COMMODORE & AMIGA nr 8



NR INDEKSU 355216
ISSN 0867-8022

Cena 10 000 zł

sierpień 1992 r.

MAGAZYN UŻYTKOWNI... ...TERÓW «COMMODORE»



- HARDWARE VIRUS PROTECTOR dla AMIGI
- JAK POŁĄCZYĆ DWIE...
- PRZENOSZENIE PLIKÓW
- DIRECTORY OPUS V3.4
- KABELKOLOGIA C-16/116/PLUS4
- FONOMASTER 64 — CZ. II
- KOLIZJE, WYP... ZDARZENIA
- AUTORUN INACZEJ
- PROGRAMOTEKA



LUCASFILM GAMES

STRIKE FLEET™

ELECTRONIC ARTS®

Gra "Strike Fleet" to właściwie pogranicze gier zręcznościowych i strategicznych, chociaż w założeniach bliżej jest tych ostatnich. Twoje zadanie polega na pomyślnym wykonaniu wielu różnorodnych zadań stawianych przed Twą flotą. Do wykonania masz 14 misji na północnym i południowym Atlantyku, Morzu Północnym i w Zatoce Perskiej. Masz okazję wykazać się umiejętnością prowadzenia zarówno jednego, samotnego statku, flotylli okrętów wojennych, jak i konwoju tankowców osłanianego przez kilka uzbrojonych kryp. Nie jest to zadanie proste, a przynajmniej — nie zawsze.

Gra polega na rozbijaniu napływających armad wroga, składających się z jednostek nawodnych i podwodnych, wspieranych przez ogień z samolotów i lądowych wyrzutni rakietowych. Twoim zadaniem jest przerobienie całego tego żelastwa na złom szybko zanurzający się w morskich głębinach. Do dyspozycji masz, większy lub mniejszy, arsenał — rakiety woda — woda bliskiego i dalekiego zasięgu, pociski przeciwlotnicze, ASROC (niezastąpiony w walce z łodziami podwodnymi), torpedy a także pociski z działek. Wszystko to w połączeniu z błyskawicznymi reakcjami Twojego mózgu powinno dać oczekiwany efekt.

Ciekawym elementem gry jest możliwość dowodzenia każdą z jednostek oddzielnie. Na bitwę można spoglądać z perspektywy okrętu flagowego, albo — np. helikoptera. Daje to nam możliwość skuteczniejszego i elastyczniejszego podejmowania decyzji. Jednak, jeżli ma się już tuzin okrętów, zaczyna się robić męczące. Być może jednak zastrzeżenie to bierze się po prostu z mojej ogólnej niechęci do wszelkiej maści symulatorów lotu, pływania, rycia i pełzania. A trzeba przyznać, że temu programowi udało się moją niechęć dość skutecznie przełamać — wystarczy przyznać, że zmusił mnie do spędzenia przed komputerem kilku godzin z dymiącą myszką i drętwiejącą prawą ręką.

Wśród scenariuszy znajdują się dwa wzorowane na amerykańskich akcjach w Zatoce Perskiej podczas operacji „Pustynna Burza". Dobrze to świadczy o aktualności gry. Z drugiej jednak strony można zauważyć jakąś manię wśród autorów — praktycznie wszystkie misje, w których pojawiają się okręty b. Związku Radzieckiego są (przynajmniej dla mnie) nie do przejścia.

Pozytywnym pomysłem jest możliwość przyspieszania i spowalniania upływu czasu. Pozwala to nam na skrócenie do minimum czasu oczekiwania pomiędzy kolejnymi falami jednostek wroga. Czas komputera można przyspieszyć nawet 128 razy, choć — jak się zdaje nie ma to większego wpływu na prędkość pracy procesora.

Instrukcja do gry jest napisana — jak to mówią — z głową. Jasno i rzeczowo wyjaśniane są po kolei wszystkie przyprawiające o zawrót głowy gadżety na tablicy rozdzielczej. Na 90 stronach instrukcja tłumaczy też jak i po co wykonujemy poszczególne misje. Podręcznik ten jest napisany w języku polskim, co — jak się zdaje — wpłynie korzystnie na atrakcyjność programu.

Bardzo elegancko autorzy rozwiązali problem zabezpieczenia przed kopiowaniem. W odróżnieniu od wielu innych znanych mi gier, nie sprowadza się on wyłącznie do niedogodności dla zwykłego użytkownika, który może np. wcale nie mieć ochoty na kopiowanie programu. Na około trzydziestu stronach instrukcji autorzy wyrysowali kilkadziesiąt różnych jednostek pływających, samolotów, helikopterów, torped i rakiet wraz z dość szczegółowymi danymi technicznymi. O te właśnie dane komputer pyta przed rozpoczęciem zabawy. Chyba jednak niesie to ze sobą niebezpieczeństwo skopiowania przez pasjonata, który wszystkie te liczby zna na pamięć...

W sumie, gra nie jest zła, mimo że dźwięk został potraktowany po macoszemu — składa się prawie wyłącznie z popiskiwania i porykiwania. Muszę przyznać, że programiści firmy Electronic Arts jeszcze raz udowodnili, że wśród producentów gier znajdują się w czołówce.

ALBERT

Dystrybutor:
IPS Computer Group, ul. Okrężna 3, Warszawa, tel. 642-2766, 642-2768.

STRIKE FLEET
KOMPUTER: Amiga 500
Rodzaj gry: strategiczna
Wymagania:



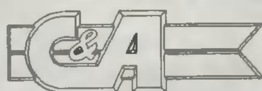


Wśród wielu listów jakie przychodzą codziennie do redakcji zwracam baczną uwagę na listy od posiadaczy Commodore 64. Ostatnio otrzymałem list, którego autor żali się, że w polskiej prasie komputerowej nastał „kompletny brak zainteresowania Commodore 64 spowodowny prawdopodobnie przekonaniem, iż maszynka ta nadaje się wyłącznie na złom".
Nie tak dawno temu miałem okazję brać udział w pewnej międzynarodowej konferencji poświęconej możliwościom zastosowania komputerów w nawigacji satelitarnej. Spotkanie to odbywało się w powszechnie znanym w Europie ośrodku szkoleniowym kształcącym młode kadry w wielu zawodach typowo lotniczych. Wyobraźcie sobie moje zdumienie, gdy na jednym ze zdjęć — a później stanowisk — zobaczyłem starego znajomego czyli... Commodore 64 z monitorem kolorowym i joystickiem.
Obrazek ten lekko mnie zaszokował, ponieważ nie do pomyślenia było, aby wspominany ośrodek miał jakiekolwiek kłopoty z nabyciem komputerów znacznie bardziej zaawansowanych technologicznie. Zamiast jednak kilkunastu Amig czy pecetów na stołach stały sobie wysłużone Commodore 64... Żądny wyjaśnień zapytałem jednego z gospodarzy do czego sprzęt ten jest wykorzystywany. Okazało się, że młodzi kandydaci na pilotów zapoznają się tu z działaniem niektórych instrumentów pokładowych. Na pytanie dlaczego właśnie Commodore 64 udzielono mi bardzo prostej odpowiedzi: ponieważ rachunek ekonomiczny wykazał, że jest to jedno z najtańszych, lecz jednocześnie efektywnych rozwiązań. Gdyby trzeba było w miejsce C-64 zainstalować pecety czy Amigi wzrosłyby na pewno koszty oprogramowania i — choć realizm byłby zapewne znacznie większy — sprawa stałaby się prawdopodobnie nieopłacalna. Ponieważ dalsze szkolenie i tak obejmowało ćwiczenia w normalnych symulatorach, gdzie uczniowie mieli do czynienia z prawdziwymi instrumentami, nie istniała żadna usprawiedliwiona potrzeba wymiany sprzętu na nowszy. Przykład ten dość mocno wspiera tezę, że wybór komputera powinien być podyktowany przede wszystkim określonymi z góry potrzebami użytkownika, a nie panującą modą, liczbą bitów procesora czy cenami. Wiadomo, że grafika Amigi pozwala na zrekonstruowanie nie tylko widoku i działania lecz nawet zadrapań na obudowie busoli; czy jednak zawsze jest to konieczne i usprawiedliwione?

KLAUDIUSZ DYBOWSKI

M·E·N·U

magazyn użytkowników komputerów «COMMODORE»



Redaktor naczelny: KLAUDIUSZ DYBOWSKI
Sekretarz redakcji: CHRISTIAN GRZENKOWICZ
Opracowanie graficzne: JOLANTA PRZEŹDZIECKA

Zdjęcia: JERZY STOKOWSKI

Stali współpracownicy: ANDRZEJ BOBEK (szef Działu Amigi)
BARTŁOMIEJ DRAMCZYK
JERZY DUDEK
MARIUSZ FERDYN
BARTŁOMIEJ KACHNIARZ
WOJCIECH KAZIMIERCZAK
PIOTR LISZEWSKI
RAFAŁ PIASEK
BARTOSZ SMAGA
RAFAŁ WIOSNA

Redakcja: ul. Wasilkowskiego 7, 02-776 WARSZAWA tel./BBS: 643-1840
Kontakt z Czytelnikami: pon-pt w godzinach 10.00-17.00
Wydawca: Spółdzielnia „Bajtek" ul. Wspólna 61 00-687 Warszawa tel./fax: 21-12-05
Skład i druk: Przedsiębiorstwo Wydawniczo-Poligraficzne „GRYF", Sp. Akc. Ciechanów
Korekta: KRYSTYNA WYDURSKA, MARIA GOŹDZIEWSKA
Nr zlecenia: 37922
Nakład: 72 tys. egzemplarzy

nowe, nowsze, najnowsze...

W Wielkiej Brytanii rozpoczęto sprzedaż nowej karty dla Amig 2000/3000, opracowanej przez firmę Commodore. Karta nazywa się 80386SX Bridgeboard i jest to sprzętowy emulator komputerów klasy PC, wyposażonych w procesory I386SX.

80386SX Bridgeboard jest pomyślany jako następca niezbyt udanego AT Bridgeboard, wszystko jednak wskazuje na to, że nie ma on żadnej z wad swego poprzednika. Karta ma pracować z częstotliwościami od 16 do 20 MHz, fabrycznie wyposażona jest w 1 MB pamięci (z możliwością rozbudowy do 8 MB) i oferuje jeden tryb graficzny, a mianowicie CGA. Fakt, że tylko jeden, nie jest jednak wielkim problemem, gdyż 386SX Bridgeboard umożliwia wykorzystywanie oryginalnych kart przeznaczonych dla „pecetów", każdy będzie więc mógł za niewielkie pieniędze kupić sobie taką kartę graficzną, na jaką będzie miał ochotę. W przeciwieństwie do AT Bridgeboard, nowy produkt firmy Commodore potrafi też w pełni wykorzystywać standardowe urządzenia do przechowywania danych Amigi, czyli jej stacje dysków, twarde dyski itd.

* * *

Znana amerykańska firma GVP wypuściła na rynek nową kartę, przeznaczoną dla Amigi 2000//3000, nazwaną IO Extender. Jak sama nazwa wskazuje, karta ta ma za zadanie rozbudowę możliwości komunikacji Amigi ze światem zewnętrznym. Zawiera ona m.in. dwa przełączalne programowo szybkie porty równoległe, jeden szeregowy (umożliwiający komunikację w obie strony), przy czym, dzięki zastosowaniu dodatkowych buforów w portach szeregowych, dane mogą być przesyłane przez nie nawet z bardzo dużymi szybkościami bez niebezpieczeństwa jakichkolwiek przekłamań.

Dodatkowo karta IO Expander może być wyposażona w interfejs MIDI, udostępniający dwie kompletne, 16-kanałowe szyny MIDI, z których na każdą składa się po pięć gniazd, w tym jedno wejściowe, trzy wyjściowe i jedno przelotowe.

* * *

Niesłychanie ciekawie zapowiadający się produkt wypuszcza na rynek firma Utilities Unlimited, znana z bardzo udanej przystawki Sybil. Nowy produkt o nazwie Emplant według swych twórców, ma umożliwiać emulację przy pomocy Amigi niemal każdego komputera.

Emplant jest emulatorem sprzętowo--programowym, przy czym hardware jest tak skonstruowany, że, przynajmniej teoretycznie, wszystko, co jest potrzebne do emulacji komputera, to jego ROM i specjalnie opracowany sterownik sprzętowej części Emplanta. Karta została bowiem tak skonstruowana, aby w kombinacji z programowanym sterownikiem umożliwiała emulację niemal dowolnego komputera. Dzięki temu, że prawie całą pracę wykonuje programowalny hardware, szybkość emulacji jest niezwykle wysoka.

W tej chwili Emplant ma być sprzedawany w konfiguracji umożliwiającej emulację komputerów Macintosh IIx, co oznacza, że dodawany będzie program emulacji tego komputera i jego ROM. Wszystkie niższa funkcje, jak na przykład timery, będzie więc emulować programowalna część sprzętowa Emplanta, zaś do grafiki, dźwięku i obsługi we/wy będzie wykorzystany hardware Amigi.

Pełny kolor emulowany jest poprzez wykorzystanie szeregu istniejących już kart graficznych, jak na przykład DCTV, HAM-E lub Firecracker 24. Układy graficzne Amigi mają obsługiwać wyświetlanie do 16 kolorów. Za emulację dźwięku odpowiedzialne będą przetworniki Amigi, jako że Mac IIx ma w tej dziedzinie możliwości niemal identyczne z Amigą. Nietypowy format zapisu dyskietek Macintosha emulowany jest za pomocą przystawki Sybil. Praca w formacie 1.44 MB wymaga zastosowania „gęstej" stacji firmy Commodore.

Dzięki elastyczności części sprzętowej Emplanta, możliwa jest w przyszłości emulacja innych komputerów — w najbliższej przyszłości mają to być Mac IIFX, Mac Quadra, Atari Mega ST oraz PC 386/486. Niestety, firma nie sprecyzowała, jak został rozwiązany problem zupełnie innego procesora w komputerach PC.

* * *

Wspomniana już firma GVP rozpoczyna w najbliższym czasie sprzedaż swej najnowszej 24-bitowej karty graficznej, przeznaczonej dla Amig 2000 wyposażonych w procesory Motorola 68030 i 68040. Karta nazywa się EGS-110/24, przy czym trzy litery oznaczają Enhanced Graphics System, czyli Rozbudowany System Graficzny.

EGS-110/24 jest w tej chwili, obok europejskiego wynalazku — Visony, jedną z najlepszych kart graficznych dostępnych dla Amigi, a właściwie to w ogóle dla „zwykłych" komputerów. Niemal wszystkie parametry jej pracy są programowalne, w tym oczywiście częstotliwość pracy (od 5 do 110 MHz), co oznacza, że karta może wyświetlać niemal dowolną rozdzielczość przy, mówiąc obrazowo ale nieprecyzyjnie, odwrotnie proporcjonalnej częstotliwości obrazu. Nie jest na przykład problemem uzyskanie rozdzielczości 1600*1300 punktów, oczywiście bez migotania. Przypomnijmy tu, że 24 bity na punkt oznaczają możliwość wyświetlenia punktu w jednym z 16.7 milionów kolorów.

EGS-110/24 sprzedawana jest z 4. lub 8. MB bardzo szybkiej pamięci RAM o czasie dostępu 25ns (czyli prawie cztery razy krótszym niż w Amidze 3000), nazwanej VRAM (od Video RAM, czyli po prostu pamięć obrazu). Nowość polega na tym, że pamięć ta rozplanowywana jest w obszarze adresowym procesora, a więc dostępna nie tylko dla karty, ale dla całej Amigi. Na tym jednak nie kończą się zalety EGS-110/24. Dotychczas dostępne karty wymagały specjalnie napisanego dla nich oprogramowania (program musiał być pisany z myślą o danej karcie), EGS-110/24 natomiast jest tak skonstruowana, że wystarczy uruchomienie specjalnego programu sterującego, aby system operacyjny (a co za tym idzie — większość programów z niego korzystających) zaczęły używać EGS-110/24 do generacji grafiki. W ten sposób, bez żadnych przeróbek programu, możemy na przykład sprawić, że DTP PageStream będzie najnormalniej w świecie pracował w rozdzielczości 1600*1300 punktów i z paletą 16.7 milionach kolorów. Równie imponująca jest sugerowana cena karty: od 2699 $ za wersję z 4 MB pamięci, do 3399 $ za EGS-110/24 wyposażoną w 8 MB.

* * *

Spadek cen sprzętu komputerowego i wszelakich przystawek nadal ma miejsce, czago dowodem może być fakt, że nowy sterownik urządzeń video (ang. media controller), skonstruowany przez firmę ARTI ma kosztować poniżej 200 $. Ma on umożliwiać sterowanie ponad czterdziestoma profesjonalnymi magnetowidami, kamkorderami czy wreszcie sprzętem wykorzystującym zapisywalne płyty, nieco podobne do kompaktowych. Ponadto przystawka potrafi sterować nawet szesnastoma urządzeniami jednocześnie. Dostępna jest ona w wersji dla Amigi, Macintosha i „pecetów".

* * *

Firma ReadySoft opracowała nową wersję swego emulatora Macintosha, nazwaną A-MAX II PLUS. Ma on postać karty instalowanej wewnątrz Amigi 2000/3000, przy czym większość pracy wykonywana jest jednak w sposób programowy. Tym niemniej zgodność programowa, jak również szybkość emulacji jest bardzo wysoka.

A-MAX II PLUS pozwala na pełne wykorzystanie osprzętu Amigi (stacji dysków, dysku twardego, portów, myszki itd.). Co ciekawsze, potrafi on za pomocą zwykłej stacji dysków Amigi zapisywać i odczytywać nietypowy format Macintosha (stała prędkość obrotowa). Karta umożliwia zastosowanie, identycznych jak w Macu, nietypowych portów szeregowych oraz zgodnego

ze standardem tego komputera interfejsu MIDI. Dzięki pełnej emulacji portu szeregowego możliwe jest wykorzystanie na przykład drukarek przeznaczonych dla Macintosha (np. Apple LaserWriter) czy przyłączenie się do sieci komputerowej. Nowa wersja oprogramowania (oznaczona 2.5) w pełni umożliwia korzystanie z najnowszego systemu operacyjnego Macintosha (V7.0.1).

* * *

Choć firma GVP znana jest głównie ze wspaniałych przystawek sprzętowych dla Amigi, ostatnio zajęła się również oprogramowaniem i, trzeba przyznać, efekty osiąga nie gorsze niż na polu „sprzętowym". Najnowszy jej twór, program graficzny Mirage, jest tym, czego wielu użytkownikom wykorzystującym Amigę do prac graficznych brakowało do szczęścia. Jest to program profesjonalny, niesłychanie rozbudowany i, zaryzykuję stwierdzenie, likwidujący różnicę pomiędzy Amigą i Macintoshem. Mirage pracuje oczywiscie w pełnych 24 bitach, czyli z paletą 16.7 milionów kolorów. Jego budowa, podobnie jak większości nowych programów dla Amigi, jest modułowa, dzięki czemu wystarczy zapisać na dyskietce nowy moduł, aby program wzbogacić o nowe funkcje.

Mirage wyposażony jest oczywiście we wszelkie funkcje, służące do „malowania" przy pomocy myszy, przy czym możliwościami znacznie przewyższa takie programy jak Deluxe Paint IV. Zresztą podobnie jak DPIV, Mirage umożliwia też tworzenie animacji, tyle że na poziomie profesjonalnym. Wbudowane funkcje umożliwiają wygodną edycję grafiki ramka po ramce, przy czym wśród funkcji jest na przykład opcja pozwalająca na uzyskanie efektów podobnych jak w filmie „Predator". Jednym słowem — pełna możliwość tworzenia animacji.

To oczywiście nie wszystkie dziedziny, w jakich musi operować dobry program graficzny. Mirage pozwala na dokonywanie wszelkich operacji na przetwarzanej grafice — mam tu na myśli wyostrzanie, transformacje kolorów, przekształcenia geometryczne, trzy rodzaje separacji kolorów (CMY, RGB i CMYK) itd. Program potrafi sterować bezpośrednio wieloma skanerami i frame-grabberami, możliwe jest również wczytywanie i zapisywanie grafiki w wielu formatach, jak IFF, JPEG, GIF, TIFF, Targa, PCX, Postscript i wiele innych.

Jako że Amiga nie potrafi standardowo wyświetlać grafiki 24-bitowej, konieczne jest korzystanie przy pracy z Mirage z kart graficznych. Tych na szczęście jest pod dostatkiem, a program potrafi sterować większością z dostępnych, (np. Impact Vision 24, DCTV, Firecracker 24, DMI Resolver). Zresztą posiadanie karty graficznej nie jest konieczne — Mirage może pracować w oparciu o standardowe tryby graficzne Amigi.

Jeszcze jedną ciekawą funkcją Mirage jest zdolność do wykorzystywania twardego dysku jako pamięci wirtualnej, dzięki czemu możliwa jest obróbka grafiki o niemal nieograniczonej rozdzielczości.

wyszukał, opracował i przetłumaczył:
ANDRZEJ BOBEK

Reprint by permission from the 5-MINUTE Weekend News Network, a *StarShip* (TM) Production on GEnie (R).

STACJA DYSKÓW 3.5" dla AMIGI

Ostatnio zauważalnym zjawiskiem na giełdach jest wzrost ilości oferowanych peryferiów do Amigi i w takiej sytuacji nawet wybredny nabywca może znaleźć coś dla siebie. Jednak chcąc kupić dodatkową stację dysków 3.5" pojawia się pytanie: którą z dostępnych na giełdzie kupić. Poszczególne rodzaje stacji różnią się pomiędzy sobą przede wszystkim rodzajem użytego napędu i wyglądem zewnętrznym. Do grona tych najlepszych, czyli do tych, w których zastosowano najlepsze napędy (np. firmy TEAC), możemy zaliczyć stację dysków 3.5" produkowaną przez spółkę AMIGA S.C. (alias Klub Komputerowy „STODOŁA").

ZESTAW, GWARANCJA I SERWIS

Firmowo napęd zabezpieczany jest za pomocą specjalnej folii chroniącej urządzenie przed uderzeniami, ale wydaje mi się, że to trochę za mało — sztywne opakowanie na pewno nie zrobiłoby złego wrażenia na kliencie.

AMIGA S.C. udziela na stację pełnej dwunastomiesięcznej gwarancji. W przypadku uszkodzenia naprawa zostanie dokonana po dostarczeniu urządzenia do producenta.

KONSTRUKCJA

Obudowę stacji wykonano z blachy pomalowanej na kolor przypominający kolor naszej Amigi. Chassis jest wykonane i solidnie i estetycznie, co w wyrobach rodzimej produkcji zdarza się wyjątkowo rzadko.

Długość przewodu łączącego stację z komputerem wynosi około 50 cm i jest ona zupełnie wystarczająca, jeżeli zamierzasz postawić stację na lub obok komputera. Jeśli jednak chciałbyś stację ustawić w innym miejscu powinieneś zadzwonić do producenta i zamówić stację z przewodem o innej długości.

Na tylnej ściance umieszczono przełącznik ON/OFF służący do włączania i wyłączania stacji, co umożliwia jej wyłączenie bez konieczności wyciągania wtyku z portu DISK DRIVE. Szczelina, w którą wsuwane są dyskietki jest zabezpieczona małą klapką, co chroni wnętrze napędu przed kurzem. Jest to dodatkowa zaleta zastosowania w stacji znakomitego napędu firmy TEAC, oraz tzw. „anticlicka" dzięki czemu praca stacji jest niezwykle cicha także bez dyskietki.

PRACA

Stację testowałem w codziennej pracy przez ponad trzy tygodnie w tym okresie nie miałem z nią



żadnych problemów. Stacja pracowała bezbłędnie ze wszystkimi dostępnymi mi programami kopiującymi (X-Copy, SuperDuper, DiskMaster, RattleCopy). Bez przeszkód przebiegała również współpraca z programem Dos2Dos — odczyt dyskietek formatowanych na różnych stacjach komputerów PC XT/AT przebiegał bezbłędnie. Bardzo miłym zaskoczeniem i jednocześnie dużą zaletą jest bardzo cicha, prawie bezszmerowa praca stacji.

NASZ WERDYKT

Opisywana tu stacja dysków 3.5" oferowana przez AMIGA S.C. jest urządzeniem na pewno godnym polecenia. Niska cena, bardzo dobry napęd, bezszmerowa praca i bezproblemowa współpraca z dostępnym oprogramowaniem gwarantują duży komfort pracy i użyteczność tego urządzenia.

BARTOSZ SMAGA

ZALETY:
+ bardzo dobry napęd
+ solidne wykonanie
+ cicha praca
+ gwarancja
+ zainstalowany „anticlick"

WADY:
– brak dobrego opakowania
– zbyt krótki przewód w standardowej wersji stacji (możliwość zmiany długości na zamówienie)

Cena (lipiec 92): 800 000 zł

PRODUCENT I DYSTRYBUTOR: AMIGA S.C. ul. Batorego 10, Warszawa, tel. 25-60-31 wew. 35.



Amiga od kuchni — cz. 4

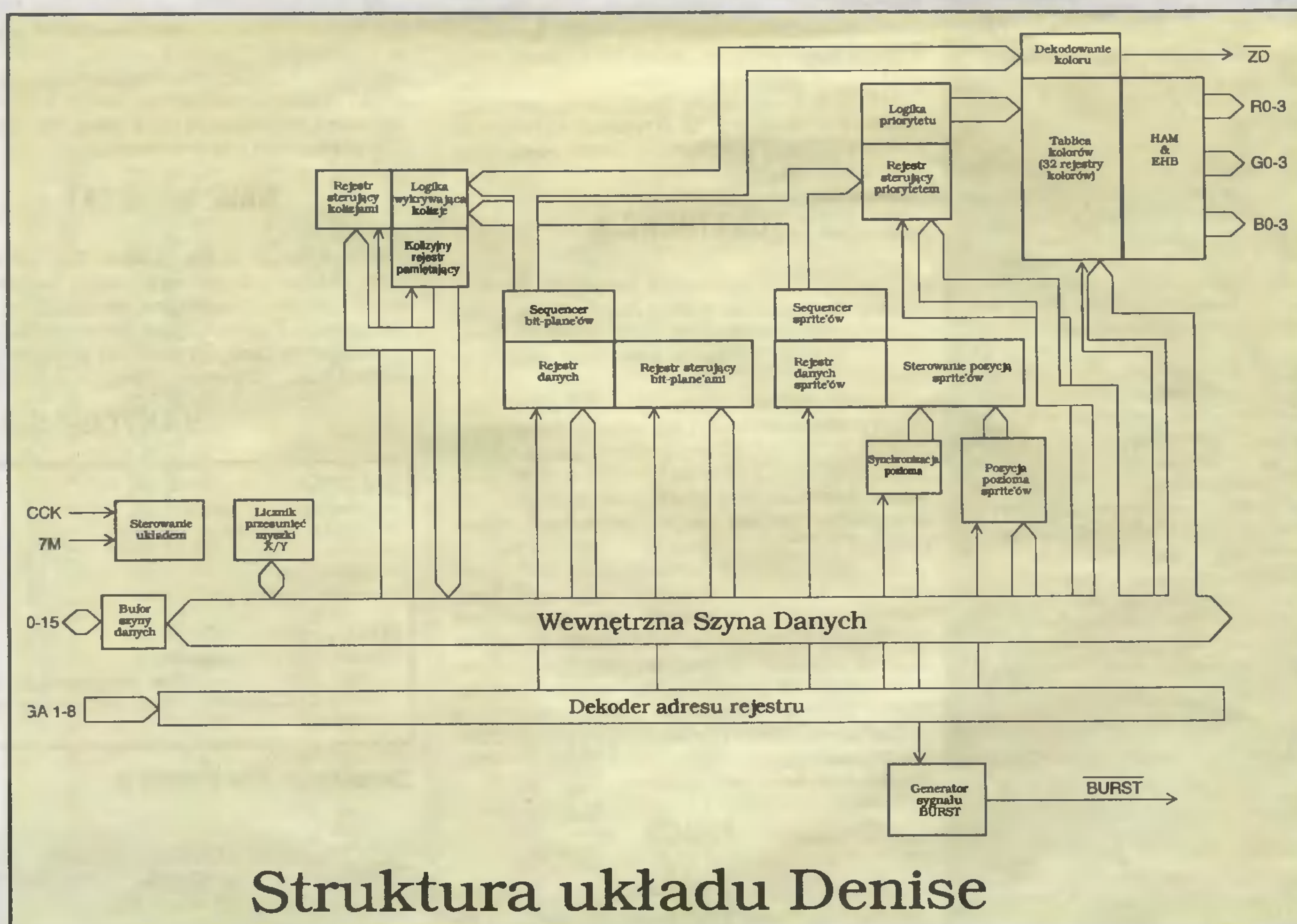
DENISE

Kolejnym specjalizowanym układem w Amidze jest układ noszący oznaczenie 8362 i nazywany Denise. Najważniejszą funkcją tego układu jest generacja obrazu (możemy go nazwać układem wizyjnym).

Pierwszą część zadania realizuje Agnus. Pobiera on z pamięci CHIP-RAM dane graficzne i zapisuje je w rejestrach Denise. To samo czyni z danymi dla sprite'ów (duszków). Rejestry Denise zawsze zawierają dane grafiki i sprite'ów dla 15 punktów (pixels) (1 bit odpowiada jednemu punktowi na ekranie, a dane w rejestrze mają długość słowa czyli 16 bitów). Te dane muszą być zamienione przez Denise na odpowiednią reprezentację sygnałów RGB. Najpierw dane graficzne są tłumaczone z 16-bitowej reprezentacji równoległej na szeregowy łańcuch danych za pomocą sekwencyjnej konwersji bit-poziom. Priorytetowa logika wybiera uporządkowane dane dla bieżącego punktu (pixel) kierując się jego priorytetem (stąd nazwa). Pobierane są zarówno dane graficzne jak i dane dla sprite'ów. Stosowanie do tych danych dekoder koloru wybiera jeden z 32 rejestrów koloru. Wartość wybranego koloru jest przesyłana jako cyfrowy sygnał RGB na wyjście. Jeśli wybrano tryb HAM (Hold-And-Modify) lub EHB (Extra-Half-Bright) to dane z rejestru koloru są odpowiednio modyfikowane zanim opuszczą układ Denise.

Dane z sekwencerów są także przesyłane do logiki zarządzającej kolizjami, gdzie sprawdzane jest występowanie kolizji (np. duszek „zetkną" się z tłem bądź z innym duszkiem). Wynik testu zostaje umieszczony w rejestrze kolizji.

Znaczenie wyprowadzeń Denise:
Szyna danych DO-D15:



Struktura układu Denise

16-bitowa szyna podłączona do szyny danych CHIP-RAM.

Szyna adresowa rejestrów RGA1--RGA8:
Ta szyna jest wyłącznie wejściem. Przy jej pomocy wybierany jest dowolny rejestr wewnątrz Denise, oczywiście za pośrednictwem dekodera adresu rejestru.

Wejścia zegarowe CCK i 7M:
Denise jest taktowany sygnałem CCK. Jest on podłączony do wyprowadzenia układu Agnus o tej samej nazwie. Sygnał na linii 7M jest sygnałem zegarowym o częstotliwości 7.15909 MHz. Denise potrzebuje tego dodatkowego sygnału do wyprowadzania poszczególnych punktów, gdyż ich częstotliwość jest większa o 3.58 MHz od sygnału CCK. W najniższej rozdzielczości jeden punkt graficzny ma czas trwania dokładnie odpowiadający sygnałowi 7M. W trybie wysokiej rozdzielczości na ekranie wyświetlane są dwa punkty w czasie jednego okresu sygnału 7M: jeden punkt-zbocze narastające, drugi-opadające. Sygnał 7M jest również sygnałem taktowania procesora 68000 i jest podłączony do jego wejścia.

Wyjścia R0-3, G0-3, B0-3, ZD i BURST:
Linie R0-3, G0-3, i B0-3 reprezentują cyfrowe wyjście RGB układu Denise. Każda z trzech składowych koloru jest reprezentowana przez cztery bity. Daje nam to 16 wartości każdej ze składowych czyli 4096 możliwych do uzyskania kolorów (16*16*16).

Zanim sygnały tę opuszczą Denise są one przesyłane poprzez bufor do trzech przetworników cyfrowo-analogowych, a następnie do portu RGB, gdzie są dostępne jako analogowy sygnał RGB. Dodatkowo mieszacz sygnałów RGB miksuje wszystkie trzy składowe a wypadkową (monochromatyczny sygnał Composite Video) podaje na wyjście VIDEO MONO. Aby prawidłowo wymieszać te składowe posługuje się on sygnałem BURST. Jest to sygnał o częstotliwości takiej samej jak CCK (3.58 MHz).

Ostatnim sygnałem wyjściowym jest sygnał ZD (Zero Detect). Przyjmuje stan aktywny (niski) w momencie, w którym punkt tła jest aktualnie wyświetlany tzn. kiedy kolor wyświetlany pochodzi z rejestru koloru o numerze 0. Sygnał ten jest wykorzystywany przez Genlock do przełączania pomiędzy sygnałem zewnętrznym (gdy ZD=0) i sygnałem wizyjnym pochodzącym z Amigi (gdy ZD=1). Sygnał ZD jest dostępny na porcie RGB.

Opracował JERZY DUDEK

HARDWARE VIRUS PROTECTOR



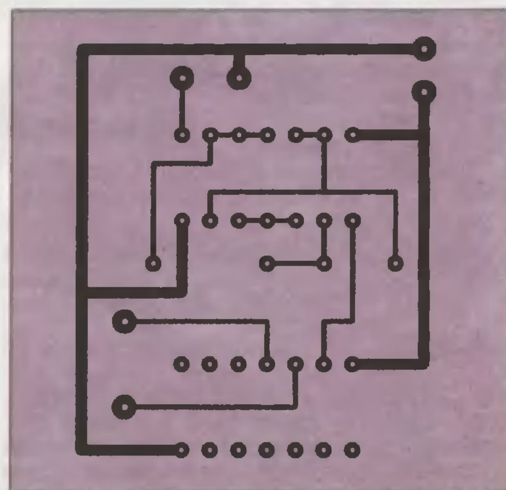
Nikt z nas chyba nie lubi wirusów. Przed tymi zapisującymi się w bootblocku może nas ustrzec opisane poniżej urządzenie. Nazwa jest może nieco na wyrost, gdyż w rzeczywistości sygnalizuje ono akustycznie fakt zapisywania czegokolwiek w bootblocku (a dokładnie: na zerowej ścieżce). Jeżeli usłyszysz sygnał w momencie, w którym żaden zapis nie ma miejsca, jest to znak, że najprawdopodobniej masz „gościa" w pamięci, i właśnie został on przeniesiony na dyskietkę.

ZASADA DZIAŁANIA UKŁADU

Wystąpienie jednocześnie stanów aktywnych (w tym przypadku niskich) na liniach DKWE (oznaczającej zapis na dyskietce) oraz TK0 (sygnalizującej pozycję głowicy nad zerową ścieżką) powoduje wystąpienie stanu wysokiego na wyjściu bramki NOR. Powoduje to wytworzenie impulsów na wyjściu generatora zbudowanego w oparciu o cztery bramki NAND. Generator stanowią pierwsze trzy bramki, czwarta jest tylko buforem separującym brzęczyk od reszty układu. Od wartości kondensatora i rezystora zależy częstotliwość generowanego tonu. Można ją określić z zależności:

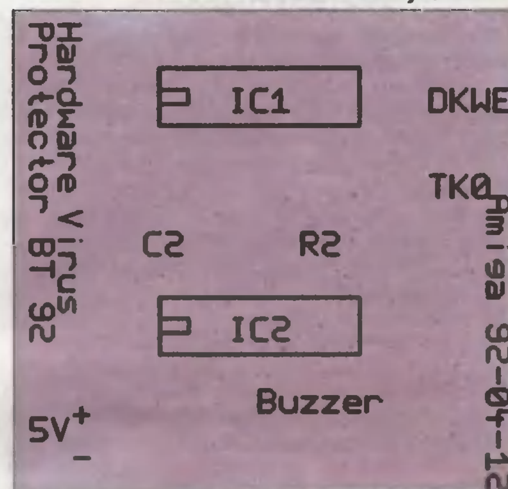
$$f=0.6/R*C$$

gdzie:
R — rezystancja opornika [Ω],
C — pojemność kondensatora [F].



Widok od spodu płytki

Widok od strony lutowania



MONTAŻ UKŁADU

Urządzenie można zmontować na płytce drukowanej przedstawionej na rysunku. Całość najwygodniej umieścić wewnątrz zewnętrznej stacji dysków, a w przypadku jej braku — podłączyć do złącza External Drive w formie małego modułu czy karty. Zastosowany w układzie brzęczyk piezoelektryczny można zastąpić innym przetwornikiem np. słuchawką telefoniczną, jednak gabaryty i głośność przemawiają za brzęczykiem.

Wykaz elementów:

Układ scalony 74LS00	1 szt.
Układ scalony 74LS02	1 szt.
Kondensator 47 nF	1 szt.
Rezystor 5.1 kΩ/0.125 W	1 szt.
Brzęczyk piezoelektryczny	1 szt.

Schemat urządzenia przedstawiam na stronie 29.

JERZY „BLUE THUNDER" DUDEK

Jak połączyć ze sobą dwie AMIGI przez RS-232

Często w listach pojawia się pytanie o możliwość połączenia dwóch komputerów ze sobą bezpośrednio (tzn. bez pośrednictwa sieci telefonicznej). Otóż połączenie takie jest możliwe i to na kilka sposobów. Dziś przedstawię, chyba najskuteczniejszy.

Schemat przewodu połączeniowego przedstawiono na rysunku. Do jego wykonania potrzebne Ci będą:

— wtyczka typu Cannon 25-stykowa żeńska (2 szt.)

— przewód siedmiożyłowy dowolnego typu

UWAGA!

NIE JEST MOŻLIWE POŁĄCZENIE ZE SOBĄ AMIG 1000 lub AMIGI 1000 Z AMIGĄ 500/ /2000 OPISYWANYM TU PRZEWODEM ZE WZGLĘDU NA INNY KSZTAŁT WTYCZKI I INNY ROZKŁAD WYPROWADZEŃ.

KONSTRUKCJA

Przewód ten umożliwia połączenie ze sobą dwóch komputerów odległych od siebie nawet o kilkaset metrów! Jak to jest możliwe? Otóż w standardzie V24 nazywanym także RS-232 logicznej jedynce odpowiada przedział od −15V do −3V, a logicznemu zeru — od +3V do +15V. Zakres od −3V do +3V jest zakresem napięć zabronionych.

Sygnał taki jest bardzo odporny na zakłócenia. W normalnej technice cyfrowej gdzie ważna jest amplituda (wartość napięcia) sygnału, łatwo może powstać przekłamanie w przypadku nałożenia się na sygnał użytkowy impulsu zakłócającego. W standardzie RS-232 ważna jest polaryzacja, a nie amplituda sygnału, przez co można go przesyłać nawet bardzo długimi, nieekranowanymi przewodami. Długość przewodu jest ograniczona tylko spadkami napięć, które nie mogą przekroczyć wartości dopuszczalnych. Dużo zależy od zastosowanych przewodów: jeśli będą one miały odpowiedni przekrój, to teoretycznie jest możliwe połączenie komputerów odległych od siebie o kilka kilometrów.

Przewód zawiera siedem żył (cyfra w nawiasie oznacza numer styku/wyprowadzenia). Funkcję każdej linii opisano poniżej:

RXD (2) — dane odbierane (Receive Data)
TXD (3) — dane nadawane (Transmit Data)
RTS (4) — część nadawcza włączona (Request To Send)
CTS (5) — gotowość nadajnika (Clear To Send)
DSR (6) — gotowość do pracy (Data Set Ready)
DTR (20) — gotowość terminala (Data Terminal Ready)
GND (7) — masa sygnałów (Signal ground/Common return)

Szeregowe przesyłanie danych po linii transmisyjnej rozpoczyna się od wysłania jednego bitu startu o wartości 0 (czyli w tym przypadku +12V) przed właściwymi bitami danych. Ponieważ linia w stanie spoczynkowym znajduje się na poziomie logicznej jedynki (−12V), to początek transmisji może być łatwo rozpoznany. Aby bity startu można było rozpoznać także i w następnych znakach, nawet gdy ostatni bit poprzedniego znaku jest zerowy, każdy znak oddziela się od następnego za pomocą jednego lub dwóch dodatkowych bitów stopu o wartości 1. Przy bitach danych jako pierwszy przesyłany jest bit najmniej znaczący. Do przekazywania znaków ASCII wystarcza tylko siedem bitów ósmy zaś jest wykorzystywany do kontroli. W celu wykrycia błędów w nadawaniu oblicza się sumę pierwszych siedmiu bitów i jeśli jest ona liczbą nieparzystą, to bit kontrolny otrzymuje wartość 1, w przeciwnym wypadku wartość 0 (nieparzystość). Czasem bit kontrolny jest negowany (parzystość).

Aby komputer odbierający nie powodował zakłóceń w nadawaniu (może być np. zajęty wydrukiem odebranych danych) i nie „zgubił" żadnego bitu stosuje się linię syganlizującą gotowość (DTR). Można do tego celu użyć także protokołu xON-xOFF. Wykorzystuje on dwa znaki sterujące z zestawu ASCII, a mianowicie znak CTRL-Q ($11) jako xON i CTRL-S ($13) jako xOFF. Odbiornik sygnalizuje za pomocą znaku xON gotowość do przyjęcia dalszych znaków, a za pomocą xOFF powstrzymuje pracę nadajnika. Zastosowanie tego protokołu ma sens wtedy, gdy chcemy połączyć dwa komputery za pomocą tylko trzech linii (żył) — stosujemy wtedy tylko linie RXD, TXD i GND. Możliwe jest również połączenie dwóch Amig przy pomocy dwóch żył (!), lecz transmisja może odbywać się wtedy tylko w jedną stronę i odbiornik musi być zawsze gotowy na przyjęcie danych.

Aby komunikacja przebiegała prawidłowo, oba komputery muszą mieć identyczne parametry transmisji i odbioru (Serial Prefs). Chodzi tu o prędkość transmisji, protokół (Handshaking), typ parzystości (Parity), liczba bitów stopu (Stop Bits), liczba bitów przesyłanych (Read/Write Bits).

W wypadku Amig redakcyjnych (łączyliśmy A500 z A2000) nie miały one problemu z przesyłaniem danych z maksymalną możliwą do ustawienia prędkością — 31250 bitów na sekundę. Pamiętaj jednak, że przewód miał w tym wypadku długość około 2 metrów i przy dalszych odległościach mogą wystąpić problemy z transmisją o tej szybkości. Radzę więc prędkość dobierać doświadczalnie.

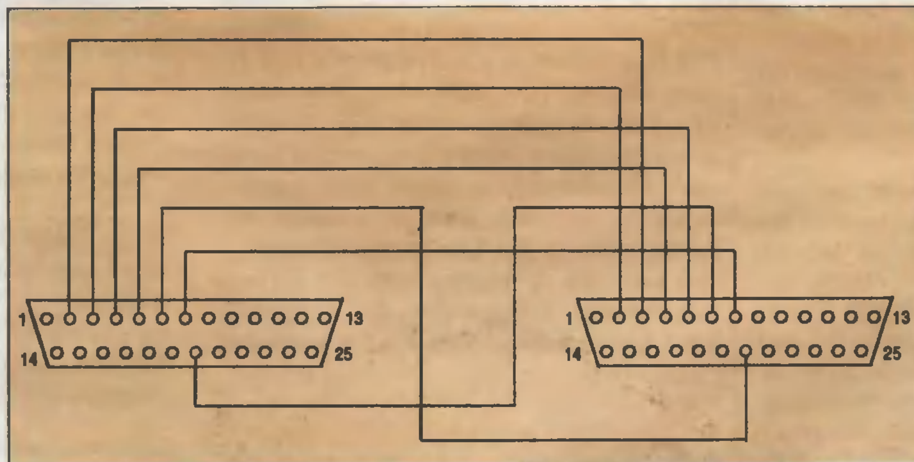
Za pomocą przedstawionego tu przewodu można podłączyć Amigę do dowolnego innego komputera wyposażonego w standardowy interfejs RS-232, np. do dowolnego PC XT/AT. Tematem tym zajmiemy się na łamach już niebawem — jak na razie połączyliśmy się z redakcyjnymi „pecetami", udało się nam również przyłączyć do C-64 i C-128.

Aby przesłać dane z jednej Amigi do drugiej należy Amidze nadającej wpisać z poziomu CLI:

copy df0: nazwapliku to ser:
Na Amidze odbierającej wpisz:

type ser:
W ten sposób dane zostaną wyświetlone na ekranie i jest to jednocześnie najszybszy test poprawności wykonania przewodu. Oczywiście kombinacji jest tu wiele, można np. napisać własny program odbierający i wysyłający dane do urządzenia ser:, może je pobierać np. z klawiatury.



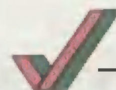


Schemat kabla łączącego dwa komputery AMIGA 500/2000 przez złącza RS-232

UWAGA:

ZE WZGLĘDU NA MOŻLIWOŚĆ USZKODZENIA OBU KOMPUTERÓW PRZY NIEWŁAŚCIWYM POŁĄCZENIU POSZCZEGÓLNYCH LINII SUGERUJĘ, ABY PODCZAS WYKONYWANIA PRZEWODU CZUWAŁ OBOK FACHOWIEC.

JERZY DUDEK



A570

Jakiś czas temu opisywaliśmy komputer Commodore CDTV, będący maszyną o całkiem nowej koncepcji, tzw. multimedia. Commodore CDTV (ostatnio sprzedawany już w wersji z klawiaturą i przemianowany na Amiga CDTV) jest maszyną opartą w 95% na konstrukcji zwykłej Amigi 500, przy czym te pozostałe 5% to stacja CD-ROM, które CDTV wykorzystuje jako podstawowy nośnik informacji. Ponieważ różnice pomiędzy tym komputerem i Amigą 500 są niewielkie, od dawna już spodziewano się wypuszczenia na rynek stacji dysków CD-ROM przeznaczonej dla A500/A500+ i umożliwiającej uruchamianie oprogramowania CDTV na tych komputerach. Tak też się stało, stacja pojawiła się ostatnio w ofercie firmy i nadano jej symbol A570.

A570 jest nieco podobna do dysku twardego A590, tak samo jak on przyłączana jest do złącza procesora z lewej strony Amigi 500. Od A590 stacja CD-ROM jest większa i nieco szersza. Podobnie jak w CDTV, A570 jest wyposażona w specjalną kasetę (CD-Caddy) w której umieszcza się płytę.

Co daje nam A570? Otóż zamienia ona naszą Amigę 500 w Commodore CDTV, dając nam możliwość użytkowania wielu wspaniałych programów, stworzonych dla tego komputera. Amiga 500 wyposażona w A570, tak jak prawdziwe CDTV, ma również zdolność odtwarzania zwykłych płyt CD, jak również tych mniej zwykłych, czyli CD+G. A570 wyposażona jest w gniazdo słuchawkowe, można ją również przyłączyć do domowego zestawu stereo.



Oczywiście A570 nie może zamienić Amigi 500 w stuprocentowe CDTV, które wyposażone jest jeszcze, na przykład, w gniazdo MIDI. Niemniej jednak wszystkie funkcje potrzebne do bezproblemowego uruchamiania oprogramowania dla CDTV są dzięki A570 zapewnione, poza tym Amiga 500 łącznie z A570 ma nad Commodore CDTV znaczną przewagę w postaci klawiatury, stacji dysków i myszki, których ten ostatni — w starej wersji — nie dysponuje.

A oto garść danych technicznych stacji CD-ROM A570:
— pojemność jednego dysku: ok. 600 MB (w przybliżeniu tyle, co 700 dyskietek)
— prędkość transmisji: 153 KB/s (tryb 1)
171 KB/s (tryb 2)
2 MB/s (tryb „burst")

Tryb odtwarzania zwykłych płyt kompaktowych:
— 8x oversampling
— pasmo przenoszenia: 20 Hz — 20 kHz
— częstotliwość próbkowania: 44 kHz
— odstęp sygnału od szumu: –102 dB
— przesłuch międzykanałowy: –92 dB
— zniekształcenia: 0.02%/1 KHz

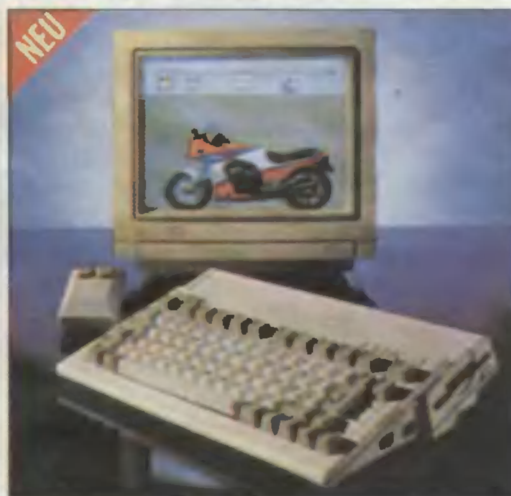
ANDY

600 czy 600HD?

Sądząc z telefonów i listów nadchodzących do redakcji C&A, nasi Czytelnicy są coraz bardziej zainteresowani kupnem Amigi 600 (HD). Jednocześnie, nie mając dostępu do materiałów prasowych, proszą nas o podanie najważniejszych parametrów tego komputera, jako że nie chcą kupować „w ciemno".

	AMIGA 600	AMIGA 600HD
Procesor	68000/16 bitowy	68000/16 bitowy
Częstotliwość zegara	7.09 MHz	7.09 MHz
Pamięć	1 MB	1 MB
(rozszerzalna maks. do)	10 MB	10 MB
Twardy dysk	(możliwy do zamontowania)	20–120 MB
Stacja dysków 3.5"	880 KB	880 KB
Memory-Card-Interface (możliwe rozszerzenie pamięci za pomocą płaskich kart RAM)	512 KB — 4 MB	512 KB — 4 MB
Rozdzielczość maks.	1280x512	1280x512
Ilość kolorów (maks.)	4096	4096
Dźwięk	4 ośmiobitowe kanały	4 ośmiobitowe kanały

Obie Amigi są wyposażone w:
— port szeregowy RS232,
— port równoległy CENTRONICS,
— wyjście RGB analogowe/cyfrowe,
— wyjście wideo (composite) PAL
— modulator TV,



— możliwość przyłączenia genlocka,
— wymiary: 36x24x7 cm.
— system operacyjny Amiga Workbench V2.0.

Memory-Cards są to płaskie karty pamięciowe wielkości karty kredytowej (przyłączalne do gniazda z lewej strony nowej Amigi). Za ich pomocą można m.in. rozszerzyć pamięć do 4 MB (Flash-RAM). Karty SRAM mogą służyć jako nośnik informacji, są bowiem wyposażone w bateryjkę podtrzymującą przy życiu zapisane dane. Natomiast karty ROM, zawierające różne (na razie jednak nie wiadomo jakie) oprogramowanie, umożliwiają podobno niezwykle szybkie wczytywanie programu do komputera. Produkowane też są specjalne karty (te płaskie oczywiście) spełniające funkcję mierników elektronicznych czy modemów.

Nowa Amiga posiada wbudowany modulator, dzięki czemu można ją przyłączyć bez żadnych przeróbek do każdego odbiornika TV. Firma Commodore twierdzi, że zdecydowana większość programów z A500 (500+) będzie „chodzić" na „sześćsetce". Aha, jeszcze jedno: A600 (HD) nie posiada EXPANSION PORT.

ARNOLD

ASEMBLER 68000:
PIERWSZE KROKI

Pierwszą rzeczą jaką zwykle chcemy uzyskać stawiając pierwsze kroki w programowaniu, jest wydrukowanie na ekranie dowolnego napisu. Prześledzimy poniżej w jaki sposób możemy tego dokonać. Użyjemy do tego celu procedur umieszczonych w pamięci ROM naszego komputera. Na początku jednak musimy zapoznać się z podstawowymi informacjami dotyczącymi procedur systemowych.

System operacyjny Amigi nazwany został przez swoich twórców KICKSTART i zajmuje około 256 KB pamięci ROM. Powstało już kilka wersji tego systemu, a najnowsza oznaczona została numerem 2.0. Programiści umieścili w nim dosłownie wszystko; począwszy od procedur obsługi pamięci i stacji dysków, a kończąc na procedurach rysujących elipsy, okręgi, czy obsługę grafiki wektorowej. Wszystkie procedury zostały pogrupowane w odpowiednie biblioteki. Część tych bibliotek znajduje się w pamięci ROM, inne muszą zostać wczytane z dyskietki. Możemy tworzyć także własne biblioteki z własnymi procedurami i zapisywać je na dyskietce.

Procedury w bibliotece umieszczone są jedna za drugą. Aby był do nich łatwy dostęp stworzono tzw. tablicę skoków. W tablicy tej umieszczono adresy pod jakimi można znaleźć poszczególne procedury. System zawsze podaje nam końcowy adres takiej tablicy, dlatego też używane przez nas wektory (określające odległość od końca tablicy do interesującego nas adresu) będą wartościami ujemnymi.

```
Procedura1 \
...          \
Procedura2    > procedury systemowe
...          /
...         /
...          \
...           \
...            > adresy procedur w
                 tabeli skoków
Adr_Procedury2 /
Adr_Procedury1/
AdresTabeliSkoków: adres tabeli skoków
                   podany jest przez
                   system
```

Jeżeli znamy położenie adresu procedury w tabeli skoków, musimy jeszcze wiedzieć gdzie w pamięci komputera tabelę tą umieszczono. Adres tabeli skoków biblioteki "EXEC" możemy odczytać pod adresem $00000004. Do innych bibliotek nie mamy takiego łatwego dostępu, muszą one zostać wcześniej otwarte. Aby otworzyć bibliotekę musimy skorzystać z procedury umieszczonej w bibliotece "EXEC" o nazwie "OldOpenLibrary": wektor tej procedury wynosi —408. Przed jej wywołaniem należy w rejestrze adresowym A1 umieścić adres początku nazwy otwieranej biblioteki. Musimy pamiętać, że nazwa musi zostać podana małymi literami i być zakończona zerem.

Spróbujmy otworzyć bibliotekę "DOS", w której umieszczone są procedury służące do: odczytu klawiatury, odczytu i z zapisu plików na dyskietce, otwierania prostych okien, drukowanie tekstu itp. Prosty program, który tego dokona powinien wyglądać w sposób przedstawiony na listingu 1.

Po wykonaniu procedury w rejestrze danych D0 system zwraca nam wartość będącą adresem tabeli skoków otwartej biblioteki. Jeżeli podana przez system wartość jest równa 0, oznacza to, że biblioteka nie została otwarta.

Teraz poznamy dwie procedury umieszczone w bibliotece "DOS": "Open" (wektor procedury = —30) i "Write" (—48). Za pomocą pierwszej procedury możemy otworzyć na ekranie proste okno, druga pomoże nam wypisać w nim nasz tekst. Wywołanie procedury "Open" wymaga podania dwóch wartości wejściowych: w rejestrze D1 musimy podać adres, pod którym umieszczone będą parametry dotyczące otwieranego okna, w D2 podajemy liczbę 1005 (określa ona rodzaj przeprowadzanej aktualnie operacji). Parametry dla okna zapisane są w następującym formacie:

CON:x/y/szer/wys/nazwa

gdzie

x/y — położenie lewego górnego rogu okna,
szer/wys — wymiary okna,
nazwa — ciąg znaków zakończonych 0, które pojawiają się na belce okna.

Po wykonaniu procedury "Open" w rejestrze D0 znajdziemy wartość, którą należy przechować (będzie nam ona potrzebna w następnych operacjach). Gdy wartość ta będzie równa zero, oznacza to, że okno nie zostało otwarte.

Teraz spróbujmy wydrukować tekst w nowo otwartym oknie. Użyjemy do tego celu procedury "Write". Przed jej wywołaniem musimy podać trzy wartości wejściowe: w D1 wartość, którą zwrócił nam system po otwarciu okna, w D2 adres początku naszego tekstu, w D3 jego długość.

Poniżej znajduje się program, który otwiera bibliotekę "DOS", otwiera nowe okno i drukuje na nim podany tekst. W tym celu wpisz program oznaczony jako listing 2.

Dobrze, otworzyliśmy okienko, wydrukowaliśmy tekst ale jak teraz to okno zamknąć? W bibliotece "DOS" znajduje się procedura "Close" (—36), która może nam w tym pomóc. Jeżeli chcemy zamknąć okno, wystarczy do rejestru danych D1 wpisać wartość, którą poda nam system po otwarciu okna i wywołać procedurę:

```
move.1 OknoHD,d1
move.1 DosBase,a6
jsr —36(a6) ; wywołanie procedury "Close"
```

Na zakończenie jeszcze jedna informacja. Jeżeli w naszym programie otwieraliśmy jakieś biblioteki, to przy powrocie do CLI, Workbencha lub asemblera, powinny one zostać zamknięte. Do zamknięcia biblioteki służy procedura "CloseLibrary" (—414) umieszczona w bibliotece "EXEC". Przed jej wywołaniem należy w rejestrze adresowym A1 umieścić wartość jaką zwrócił nam system po otwarciu biblioteki. Poniżej znajduje się fragment programu, który zamknie otwartą wcześniej bibliotekę "DOS".

```
move.1 $4,a6 ; w A6 adres tabeli skoków "EXEC"
move.1 DosBase,a1
jsr —414(a6) ; zamknięcie biblioteki
```

BARTOSZ SMAGA

LISTINGI — patrz str. 31



O PRZENOSZENIU DANYCH SŁÓW KILKA (cz. 1)

Czego by nie mówić o jakości komputerów zgodnych z IBM PC, nie da się ukryć, że są one wszechobecnym standardem. Można je spotkać w prawie każdym zakątku globu, w szkole, w pracy, słowem — wszędzie. Nic więc dziwnego, że często zachodzi konieczność wymiany danych pomiędzy Amigą a „pecetami"

Dane te mogą mieć różną formę — grafiki, tekstu, zbioru bazy danych czy jedną z setek innych. Zawsze jednak pierwszy etap to przeniesienie samego pliku, dopiero potem przychodzi czas na ewentualną konwersję. Chciałbym więc rozpocząć cykl od przedstawienia najbardziej znanych programów służących do mniej lub bardziej wygodnego przenoszenia danych pomiędzy Amigą a popularnymi „pecetami", jak również, niejako przy okazji, Atari ST, którego format zapisu różni się bardzo nieznacznie od stosowanego w „pecetach".

Pierwszym służącym do tego programem, dziś już dość leciwym, był *Dos2Dos* firmy Central Coast Software. Rzadko się zdarza, by najstarszy program był najlepszy i *Dos2Dos* nie jest tu wyjątkiem, czego nie zmienia nawet fakt, że cały czas pojawiają się jego nowe wersje — zmiany dotyczą jedynie szczegółów. Jest on niewygodny w obsłudze, ma bardzo ograniczone możliwości, a przy tym jest bardzo wolny. Oczywiście nie brak mu również kilku zalet — jedną z nich jest bezawaryjna praca z dyskietkami w formacie 360 KB, jak również możliwość konwersji zbiorów ASCII (Amiga jako kod końca linii stosuje jeden bajt, IBM — dwa). Dla mniej doświadczonych amigowców zaletą może być jeszcze fakt, że D2D nie korzysta z żadnych dodatkowych plików, jak device'y czy handlery, które trzeba instalować w katalogach systemowych. Na tym jednak zalety *Dos2Dos* się definitywnie kończą.

Następne programy do przenoszenia danych pomiędzy Amigą a PC miały już diametralnie zmienioną koncepcję. Zasada ich działania jest wbrew pozorom niezbyt wyszukana, jednak efekt końcowy jest po prostu świetny — programy te, mówiąc obrazowo, „uczą" system operacyjny Amigi rozpoznawania dysków w formacie MS-DOS, dzięki czemu niemal całe oprogramowanie korzystające z systemu operacyjnego jest w stanie najnormalniej w świecie odczytywać i zapisywać dyski „pecetowe". Dowcip polega więc na tym, że do operacji na dyskietkach w formacie IBM PC możemy użyć KAŻDEGO programu w rodzaju Disk Mastera czy Directory Opus. Poza tym możemy, najzwyczajniej w świecie, korzystać z tych dyskietek przy normalnej pracy, na przykład z programem graficznym, edytorem tekstów czy DTP.

Pierwszy z tego typu programów to *CrossDOS*, produkt komercyjny firmy Consultron. Jego działanie przejawia się tym, że do spisu urządzeń systemowych (czyli na przykład stacji dysków) dodaje on nowe, o symbolu DIx:, gdzie x jest numerem stacji dysków. Jeśli więc chcemy używać dysków w formacie MS-DOS albo Atari ST, to wystarczy zamiast, dajmy na to „DFO", użyć „DIO:" i sprawa jest załatwiona. Zresztą istnieje też możliwość zmiany oznaczenia DIx: na jakiekolwiek inne, na przykład „A:" lub „B:".

CrossDOS ma sporo zalet: dużo wyższą od D2D prędkość działania, możliwość włączenia konwersji ASCII, współpraca z dyskietkami 360 KB. Do jego wad należy bezsprzecznie skomplikowany proces instalacji: nie dość, że trzeba zainstalować kilka składających się na niego programów, to dodatkowo musisz jeszcze umieścić w katalogach systemowych device i handler, zmienić tzw. Mountlist, po czym trzeba jeszcze uruchomić program zmieniający wektory procedur obsługi stacji dysków. Na zakończenie „podłączamy" urządzenie „DIx:" za pomocą polecenia Mount. Jak widać z przytoczonej listy, dla mniej doświadczonego użytkownika Amigi procedura instalacyjna tego programu może okazać się istnym torem przeszkód. Starsze wersje programu (podobnie zresztą było z *Dos2Dos*) odmawiały też kategorycznie pracy pod kontrolą systemu operacyjnego w wersji 2.0, jak również miały zwyczaj odnajdywania na dyskach nie istniejących błędów.

CrossDOS był programem nowym, wykorzystującym niesamowitą elastyczność systemu operacyjnego Amigi, dzięki czemu zdobył sobie na początku spore uznanie. Jego kariera skończyła się jednak z chwilą, gdy pojawił się *MultiDOS*. Autorem tego programu jest Norweg, Kjell Didrigsen, sam zaś program jest przeznaczony do **bezpłatnego** rozpowszechniania. *CrossDOS*, mimo że jest programem komercyjnym i wcale nie takim znowu tanim, ustępuje *MultiDOSowi* pod każdym względem, może tylko z wyjątkiem faktu, że nie radzi sobie z formatem Atari ST (a przynajmniej ja nie umiałem go do tego zmusić). Nie pierwszy to już raz się tak dzieje, że produkt pracy zapaleńców przewyższa jakością wytwory profesjonalistów...

Instalacja programu *MultiDOS* na dysku nie jest niestety wiele prostsza niż *CrossDOSa*. Choć trzeba kopiować device i handler, to jednak nie zachodzi już na przykład konieczność modyfikacji Mountlist. Sam program uruchamia się też dużo prościej, z pomocą tylko jednego polecenia. Po uaktywnieniu MultiDOS system operacyjny rozpoznaje dwa nowe formaty dyskietek: standardowy MS-DOS i drugi, będący mieszanką formatu Amigi i „peceta". Zastosowanie pierwszego formatu jest chyba oczywiste, drugi zaś też może się przydać — na dyskietce mieści się o kilkadziesiąt KB więcej danych, katalog odczytywany jest zawsze błyskawicznie (jak w PC), zaś praca z plikami również odbywa się dużo szybciej (w skrajnych przypadkach nawet o 80%). Szybkość pracy jest zresztą jednym z podstawowych atutów tego programu — *MultiDOS* działa około pięć razy szybciej niż *Dos2Dos* i nieco ponad dwa razy szybciej niż *CrossDOS*.

Następną zaletą *MultiDOSa* jest sposób, w jaki realizuje swe zadanie. Otóż nie tworzy on sztucznie nowego urządzenia, lecz przejmuje po prostu kontrolę nad wszystkimi stacjami dysków. Po włożeniu dyskietki w formacie Amigi *MultiDOS* przekazuje kontrolę nad nią do systemu. Jeśli jednak dyskietka jest zapisana w jednym z dwóch pozostałych formatów — kontrolę przejmuje *MultiDOS*. Tak więc, z punktu widzenia użytkownika, nic się w sposobie korzystania ze stacji nie zmienia — cały czas operujemy nazwą DFx:, resztę załatwia już program.

Omówienie *Dos2Dos*, *CrossDOS* i *MultiDOS* nie wyczerpuje oczywiście tematu, dla Amigi istnieje bowiem jeszcze kilka programów o podobnych zastosowaniach (np. *MSH*, *ADOS*), niemniej jednak opisywanie ich wszystkich byłoby bez sensu, gdyż w większości są one bardzo zbliżone do *CrossDOSa*, a równocześnie gorsze od *MultiDOSa*.

Za miesiąc opiszę ciąg dalszy procesu, a mianowicie programy służące do przenoszenia grafiki, a dokładniej mówiąc — do jej konwersji.

ANDRZEJ BOBEK

DIRECTORY OPUS V3.4

Na pewno każdy z Was po wstępnej fascynacji grami zaczął zajmować się bardziej poważnymi zastosowaniami naszego wspaniałego komputera. Jedną z podstawowych operacji stało się przenoszenie plików z dyskietki na dyskietkę, a także ich edycja. Większość z Was korzysta z programów takich jak Disk Master, File Master, Sid itp. Ja jednak chciałbym przedstawić Wam inny program tego typu — Directory Opus.

Jest to jeden z najbardziej zaawansowanych programów w tej grupie napisanych dla Amigi. Przy Directory Opus programy wymienione poprzednio są tylko prostymi ulepszeniami w stosunku do programików zawartych na dysku systemowym. Niestety program ten mogą docenić tylko posiadacze co najmniej 1 MB pamięci RAM i najlepiej 2 stacji dysków lub dysku twardego. Należy również dodać, że program ten jest programem komercyjnym i kosztuje około 50 dolarów australijskich, lecz są to jednak na pewno dobrze wydane pieniądze. Użytkownikom innych komputerów (mam tu na myśli głównie pecety) pierwsze wrażenia będą zwykle nasuwały nieodpartą myśl, że pierwowzorem Directory Opus jest pakiet Norton Commander. Rzeczywiście wrażenia te są podobne, natomiast pod względem możliwości Opus wyprzedza bardzo znacznie swojego pradziada.

Directory Opus pozwala użytkownikowi na dokonanie na pliku bądź plikach dowolnej operacji przy użyciu wyłącznie myszki (wszystkie funkcje są również dostępne za pośrednictwem klawiatury). Program zawiera też opcję HELP, bardzo pomocną dla początkujących.

Podczas pracy programu ekran monitora podzielony jest na cztery części. Widzisz dwa okna, w których dokonujesz wszystkich operacji. Poniżej zlokalizowano przełączalne banki gadżetów, którym możesz przypisać dowolne funkcje takie jak COPY, MOVE, DELETE, SHOW, PLAY, EDIT i wiele innych (standardowo jest ich około 40). Dalej znajduje się linia informacyjna, zawierająca aktualny czas i datę, wyświetlająca dane o zajętości pamięci (z podziałem na CHIP i FAST) oraz stopień obciążenia procesora wyrażony w procentach. Ponad oknami znajduje się pięć rozwijanych menu, z których dwa są już częściowo wykorzystane przez opcje FORMAT, INSTALL, QUIT, ICONIFY itp. Operacje wykonywane Directory Opus można podzielić na kilka grup:

1. Standardowe operacje wykonywane na plikach, znane z innych programów tego typu, tj. COPY, MOVE, DELETE, RENAME, MAKEDIR, HUNT, PROTECT, itd.

2. Operacje dyskowe, czyli FORMAT, INSTALL, INFO, ADDBUFFERS, itd.

3. Operacje typu SHOW, PLAY, READ, EDIT, umożliwiające:

— obejrzenie dowolnego obrazka wraz z automatycznym jego pośrodkowaniem na środku ekranu (możesz sobie wybrać inny program służący do tego celu, jeśli nie odpowiada Ci wbudowany),

— obejrzenie dowolnej ikony czy krojów czcionki znajdujących się na dyskietce (dysku),

— uruchomienie modułu muzycznego, zapisanego w formacie Soundtrackera, czy dowolnego pliku (sampla), oczywiście, tak jak w poprzednim przypadku, możesz sobie wybrać inny program służący do tego celu,

— odczytanie dowolnego pliku tekstowego (nawet „spakowanego" Power Packerem) wybierając, czy ma się to odbyć normalnie, w kodzie szesnastkowym, czy ANSI,

— edycję danego pliku (np. startup-sequence) wybranym przez Ciebie dowolnym edytorem tekstu,

4. Operacje ustalane są w dowolny sposób przez użytkownika. Z poziomu Directory Opus można uruchomić dowolny inny program z uwzględnieniem, jak ma się to odbyć:

— osobny task (zadanie), czy nie,
— ekran Opusa może być przeniesiony za ekran systemowy,
— otwarcie osobnego okna, w którym dany program będzie pracować,
— możliwość zamknięcia Opusa przed uruchomieniem naszego programu. Jego „ikonifikacja" (zastąpienie ekranu małą ikoną, po wybraniu której Opus uaktywni się ponownie).

Możemy również z poziomu Opusa uruchamiać programy archiwizujące, czyli Lharc, LZ, ZOO. Powiecie, że to nic nowego, przecież takie możliwości ma wiele innych programów choćby i Disk Master. Prawda, lecz tutaj odbywa się to w sposób szalenie wygodny. Wywołania tych programów można zainstalować sobie w dowolnym miejscu zależnie od wymagań użytkownika (gadżet czy menu). Również niezbędne parametry można przekazywać w czasie ich uruchamiania, nie tylko w czasie ich instalacji. Innymi słowy wszystko zależy tu od użytkownika...

Również dla osób zajmujących się przesyłaniem plików pomiędzy Amigą i „pecetami" czy Atari program ten będzie bardzo pomocny. Umożliwia on wykonywanie konwersji w sposób wygodny i przyjemny. Wystarczy zaopatrzyć się w program CrossDos v4.0 (komercyjny) czy MessyDos (shareware) i zainstalować go. Od tej chwili wszystkie operacje zamiany formatów mogą być dokonane tylko i wyłącznie za pomocą myszki (w przeciwieństwie do programów takich jak Dos2Dos).

Wraz z programem Directory Opus dostarczane są programy pomocnicze:

— konwerter umożliwiający zamianę konfiguracji programu z formatu używanego w poprzednich wersjach Opusa, na format aktualny. Nie zmusza to użytkownika do ustawiania od początku swojej „starej" konfiguracji — konieczna jest jedynie zamiana starszego formatu na nowszy.

— program ConfigOpus, który powinien być zapisany na dysku wraz z Opusem (w katalogu głównym katalogu lub w katalogu C).

Program ten pozwala nam na zmianę konfiguracji. Wystarczy teraz wczytać Opusa, a następnie z menu Project wybrać opcję „Configure". Po chwili ukaże się nam menu programu konfiguracyjnego i możemy szaleć do woli. Poza zmianą ogólnego wyglądu programu (umiejscowienie okien, banku gadżetów) możemy zmienić dosłownie wszystko. Od takich drobiazgów jak ilość pokazywanego wolnego miejsca na dysku (w blokach, bajtach, kilobajtach, lub procentach — do wyboru) czy zmiany rodzaju zegara (12. lub 24. godzinny), poprzez zmiany układu gadżetów a także ich nazw, przypisanych im funkcji i liczby banków, do zmian czcionek wykorzystywanych przez program (osobno dla okien, dla gadżetów, i rozwijanych menu). Możesz również zmienić liczbę kolorów ekranu Opusa (2, 4, 8 lub 16) i ich ustawienia (z doświadczenia wiem, że najwygodniejsze jest pozostawienie tej wartości bez zmian — czyli 4).

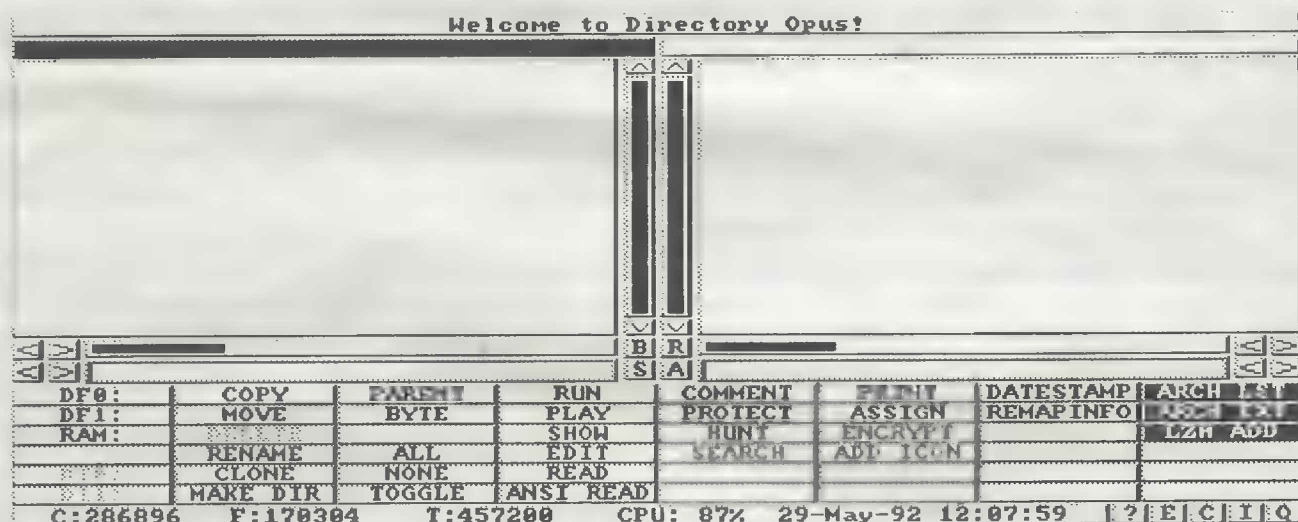
Jedną z największych zalet Opusa jest rozpoznawanie rodzaju pliku (tzn. obrazek, czcionka czy np. tekst ASCII), dokonywane automatycznie przez komputer. Na czym to polega? Jeżeli po odczytaniu katalogu dyskietki nie będziemy mogli stwierdzić, jakiego rodzaju jest jakiś plik, to wystarczy dwukrotnie „kliknąć" na nim myszką, a Opus dokona już reszty. W zależności od rodzaju pliku na ekranie pojawi się obraz czy tekst, usłyszysz muzykę (gdy był to moduł czy sampling) itp.

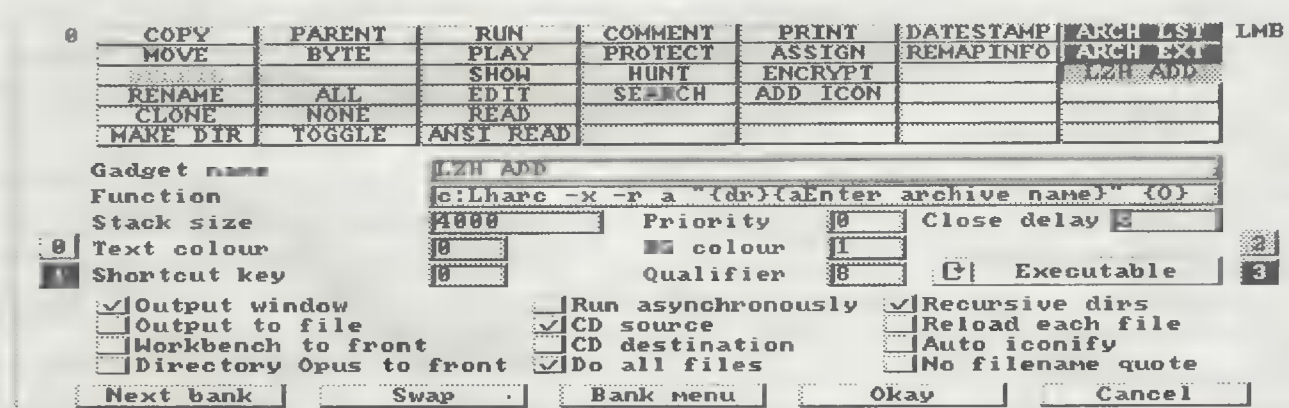
Więcej, mamy możliwość takiego ustawienia konfiguracji, że w przypadku napotkania na plik z rozszerzeniem na przykład .LZH (typowym dla archiwów tworzonych za pomocą programu LHARC) program pokaże nam automatycznie jego zawartość lub spróbuje taki plik „rozpakować" (oczywiście możemy nauczyć Opusa rozpoznawania innych rozszerzeń np. .LHA, .DMS, .ZOM, czy MOD.). Identyfikacja plików nie następuje tylko na podstawie rozszerzenia — Opus może rozpoznawać je na podstawie dowolnie wybranego, stałego dla danego typu plików ciągu bajtów.

Po wyliczeniu najważniejszych możliwosci Opusa (oczywiście nie wszystkich, gdyż wyliczenie i nawet pobieżne opisanie wszystkich jego możliwości zajęłoby co najmniej pół numeru C&A) nadszedł czas na jego wady. Jest ich naprawdę niewiele. Do najważniejszych można zaliczyć zawieszanie się programu w najmniej oczekiwanym momencie (na szczęście zdarza się to bardzo rzadko, niemniej jednak się zdarza). Niestety cały pakiet ma niezbyt „szczupłe" zapotrzebowanie tak na pamięć RAM jak i na miejsce na dysku. Wraz z potrzebnymi mu bibliotekami zajmuje on prawie 0.5 MB pamięci, co przy typowej konfiguracji (1MB RAM) zmusza użytkownika do korzystania z dwóch stacji dysków lub dysku twardego, albo rozszerzenia pamięci.

Duże rozmiary programu czynią Directory Opus programem mniej popularnym od innych tego typu. Należy jednak dodać, że Directory Opus jest programem w pełni profesjonalnym, a profesjonalizm na Amidze zaczyna się od 3 MB pamięci i dysku twardego. Spore rozmiary programu dają pojęcie o bogactwie oferowanych możliwości; warto również wiedzieć, że na jego objętość miał wpływ język w jakim Directory Opus został napisany (C).

HIGHTOWER





COMPLEX INTERFACE ADAPTER #1

— DRUGIE DANIE

W drugiej części opisu układu CIA#1 nasmażyło się trochę rejestrow. Ponieważ w ostatnim odcinku strasznie zamarudziłem nad pierwszymi czterema rejestrami (mam nadzieję, że nikt nie umarł z przejedzenia ?!?), tym razem solidna pigułka informacyjna — dalsze 14 rejestrów CIA#1.

56324 ($DC04) TIMALO CIA#1+4
Młodszy bajt timera A

56325 ($DC05) TIMAHI CIA#1+5
Starszy bajt timera A.

56326 ($DC06) TIMBLO CIA#1+6
Młodszy bajt timera B.

56327 ($DC07) TIMBHI CIA#1+7
Starszy bajt timera B.

Tak. Znowu cztery rejestry. Odczyt ktoregoś z timerów (w postaci: TIMALO + 256*TIMAHI) okresla aktualną wartość wybranego licznika, który nie robi nic więcej poza ciągłym odliczaniem od zadanej wcześniej wartości do 0. Tak więc po wpisaniu do timera i włączeniu go (o tym za chwilę), zadana wartość będzie zmniejszana o 1 w każdym cyklu zegarowym mikroprocesora. Dla systemu TV PAL (a więc wszystkich komputerów europejskich) częstotliwość zegara wynosi 985250 Hz, dla NTSC natomiast (USA, Kanada) — 1022730 Hz. Każde zmniejszenie licznika trwa więc około jednej milionowej części sekundy. Gdy timer A zliczy do 0, zostanie ustawiony bit 0 rejestru CIAICR i, o ile przerwania z tego timera są odblokowane, — zostanie wywołana procedura wskazana przez wektor IRQ, natomiast timer A prześle dane (0 lub 1) do bitu 6 rejestru CIAPRB (jesli CIAPRB jest ustawiony jako wyjście timera A). Następnie timer A zostanie zatrzymany lub ponownie zacznie odliczać od ustalonej wcześniej wartosci.

Podobnie działa timer B, z tym, że standardowo nie wywołuje on przerwań przy przejściu przez 0. Informacja o osiągnięciu 0 w tym timerze wpisywana jest do bitu 7 CIAPRB.

Timer A poza odliczaniem cykli zegara, może zliczać zewnętrzne impulsy (linia CNT1, złącze 4 portu użytkownika). Timer B może także zliczać przejścia przez 0 timera A. W ten sposób można połączyć oba timery w jeden 32-bitowy, który może odliczać odcinki czasu do ok. 70 minut.

Normalnie timer A jest wykorzystywany przez system operacyjny do generowania przerwań IRQ. Wraz z timerem B obsługuje magnetofon (synchronizacja transmisji danych). Timer A ustawiony jest na działanie ciągłe, poczynając od wartości 17045 (TIMALO=147, TIMAHI=66). Cały cykl timera A trwa 17045/ /985250 (NTSC: 17045/1022730), co w przybliżeniu wynosi 1/60 s.

56328 ($DC08) TODTEN CIA#1+8
Rejestr zegara czasu rzeczywistego (dziesiętne części sekund). Bity 0–3: dziesiętne części sekundy (w kodzie BCD), bity 4–7: niewykorzystane.

56329 ($DC09) TODSEC CIA#1+9
Rejestr zegara czasu rzeczywistego (sekundy).
bity 0–3: druga cyfra sekund (BCD),
bity 4–6: pierwsza cyfra sekund (BCD),
bit 7: niewykorzystany.

56330 ($DC0A) TODMIN CIA#1+10
Rejestr zegara czasu rzeczywistego (minuty)
bity 0–3: druga cyfra minut (BCD),
bity 4–6: pierwsza cyfra minut (BCD),
bit 7: niewykorzystany.

56331 ($DC0B) TODHRS CIA#1+11
Rejestr zegara czasu rzeczywistego (godziny i pora dnia)
bity 0–3: druga cyfra godzin (BCD),
bit 4: pierwsza cyfra godzin,
bity 5–6: niewykorzystane,
bit 7: wskaźnik pory dnia:
1 — po południu (PM),
2 — przed południem (AM).

Witajcie w krainie zegarków! CIA#1 poza dwoma timerami ma wbudowany zegar czasu rzeczywistego, zapisany w kodzie BCD (Binary Coded Decimal). Jest to zegarek z „alarmem", wywołującym przerwanie w momencie, na który ustawiłeś „alarm".

Format zapisu zegara w kodzie BCD został przyjęty po to, żeby łatwiej było przetwarzac wskazania zegara na kody ASCII. W formacie tym bajt podzielony jest na dwa nibble (nibble to inaczej cztery bity) reprezentujące wartości dziesiętne od 0 do 9 (mimo, że w jednym nibble można przechowac wartości od 0 do 15). Na przykład 32 minuta reprezentowana jest przez binarne wartości 0011 (3) i 0010 (2). Wartośc całego bajtu (na który składa się starszy i młodszy nibble) to 35, ale w kodzie BCD to ni mniej ni więcej tylko 32! Ten sposób zapisu dotyczy wszystkich rejestrów zegara czasu rzeczywistego. W rejestrze TODHRS bit numer 7 określa porę dnia, z kolei na pierwszą cyfrę godzin przeznaczono tylko 1 bit (o numerze 4).

Zegar czasu rzeczywistego jest uruchamiany w momencie włączenia komputera. Nie zatrzymują go ani operacje we/wy, ani inicjalizacja (reset) systemu.

UWAGA:
Odczyt rejestru godzin blokuje uaktualnianie wskazań zegara (zegar jednak pracuje dalej) — chodzi tutaj o uniknięcie błędnego odczytu zegara. Ponowne odnowienie wskazań zegara następuje po odczytaniu rejestru TODTEN.

Jeśli w czasie zapisu rejestrów zegara bit 7 rejestru CIAPRB ma wartość 0, to następuje ustawienie zegara, jeśli bit ten będzie ustawiony (1) — możesz ustawić „alarm". Jedyną widoczną funkcją zegara czasu rzeczywistego jest instrukcja RND(0), która wykorzystuje rejestry TODTEN i TODSEC do generowania liczb pseudolosowych. Nie jest to jednak zbyt dokładne „losowanie". Wynika to z formatu zapisu tych rejestrów w kodzie BCD (niektóre wartości nie pojawiają się). Działanie tego zegara można obejrzeć na przykładzie programu zamieszczonego w artykule „Zegar TOD" (C&A 04/92).

Zegar czasu rzeczywistego jest znacznie dokładniejszy niż zegar znajdujący się w komórkach 160–162 (tak zwany jiffy clock), ustawiany i odczytywany za pomocą zmiennej TIME$. Do głównych wad tego ostatniego należy zatrzymywanie pracy na czas operacji we/wy.

56332 ($DC0C) CIASDR CIA#1+12
Port szeregowy

Rejestr ten umożliwia szeregową transmisję danych (czytaj bit po bicie, względem wieku: najstarszy na pierwszy ogień). Może służyć zarówno jako nadajnik (bit 6 rejestru CIACRA ustawiony), jak i odbiornik (stan logiczny 0 bitu 6 rejestru CIACRA). Przy pracy jako nadajnik dane wpisane do CIASDR zostaną wysłane w postaci ciągu bitów na linię SP1 USER PORT (złącze 5). Pełny cykl timera A okresla czas w którym przesyłany bit będzie utrzymywany na linii SP1. Jeśli timer A pracuje w trybie ciągłym, to transmisja danych następuje zaraz po wpisaniu ich do CIASDR. Bity są transmitowane z częstotliwością dwukrotnie mniejszą niś częstotliwość przechodzenia timera A przez 0. Na linię CNT1 (4 złącze USER PORT) przesyłana jest informacja o przesłaniu kolejnego bitu. Po przesłaniu całego bajtu następuje oczekiwanie na wpisanie kolejnego do CIASDR.
Podczas pracy jako odbiornik bity danych odczytywane są z linii SP1 (złącze 5 USER PORT). Sygnał na linii CNT1 określa czy należy odczytać kolejny bit. Po odczytaniu całego bajtu zostaje zgłoszone przerwanie żądające odczytu danych z CIASDR. C-64 nie wykorzystuje tego rejestru. Wszystkie transmisje szeregowe przeprowadzane są za pomocą portu wewnętrznego procesora (komórka $0001).

56333 ($DC0D) CIAICR CIA#1+13
Rejestr sterujący przerwaniami.

bit 0: odczyt : 1 — timer A zliczył do 0,
0 — timer A nie zliczył do 0,
zapis : 1 — włączenie przerwań z timera A,
0 — wyłączenie przerwan z timera A,
bit 1: odczyt : 1 — timer B zliczył do 0,
0 — timer B nie zliczył do 0,
zapis : 1 — włączenie przerwan z timera B,
0 — wyłączenie przerwań z timera B,
bit 2: odczyt : 1 — zegar czasu rzeczywistego wskazuje czas ustawiony w „alarmie",
0 — zegar czasu rzeczywistego nie osiągnął „alarmu",
zapis : 1 — włączenie przerwań „alarmu",
0 — wyłączenie przerwań „alarmu",
bit 3: odczyt : 1 — port szeregowy nadał cały bajt
0 — port szeregowy jeszcze nie nadał całego bajtu,
zapis : 1 — włączenie przerwan z portu szeregowego,
0 — wyłączenie przerwań z portu szeregowego,
bit 4: odczyt: 1 — został przesłany sygnał na linię FLAG,
0 — sygnał nie został przesłany na linię FLAG,
zapis : 1 — włączenie przerwań z linii FLAG,
0 — przerwania z linii FLAG wyłączone,
bity 5–6 : niewykorzystane,
bit 7: odczyt : 1 — źródłem przerwania jest układ CIA#1,
0 — przerwanie pochodzi spoza układu,
zapis : 1 — bity wpisane jako "1" będą ustawiać bity rejestru CIAICR,
0 — bity wpisane jako "1" będą zerować bity tego rejestru.

Rejestr CIAICR steruje wszystkimi możliwymi przerwaniami układu CIA#1. Jest ich pięć: timer A, timer B, zegar czasu rzeczywistego, port szeregowy, oraz linia FLAG. Oba timery wywołują przerwanie (o ile jest włączone), po zliczeniu do zera. Zegar czasu rzeczywistego generuje przerwanie, gdy osiągnie czas „alarmu" Port szeregowy powoduje przerwanie po nadaniu całego bajtu danych. Ostatnie źródełko przerwań to sprzętowa linia FLAG, dołączona do linii READ magnetofonu. Powoduje przerwanie, gdy pojawi się na niej stan niski.
Aby przerwanie zostało obsłużone, musi zostać najpierw włączone. Wartosci wpisywane do CIAICR są uzależnione od bitu 7. Jeżeli jest on wyzerowany, to wszystkie pozostałe bity wpisywane jako "1" będą traktowane jako "0" i na odwrót. Dlatego też instrukcja POKE 56333,127 wyłącza wszystkie przerwania, a POKE 56333,129 włącza przerwania z timera A. Próba wykonania tych instrukcji w trybie ekranowym (tzn. bez programu) spowoduje odłączenie klawiatury. Odczyt rejestru CIAICR pozwala określić co było przyczyną wywołania przerwania (jeden z bitow zostanie ustawiony). Na przykład jeśli zostanie ustawiony bit 0, to wiadomo, że timer A odliczył do 0. O ile włączone są przerwania z tego timera zostanie wywołana procedura przerwań IRQ, oraz ustawiony bit 7 CIA-ICR (przerwanie pochodzi z CIA#1). **Uwaga! Odczyt tego rejestru kasuje jego zawartość.**

56334 ($DC0E) CIACRA CIA#1+14
Rejestr sterujący A.
bit 0 : 1 — włączenie timera A,
0 — wyłączenie timera A,
bit 1 : 1 — wyprowadzenie danych na bit 6 CIAPRB, gdy timer A zliczy do 0,
0 — wyprowadzanie danej wyłączone,
bit 2 : 1 — sposób wyprowadzania danych: zmiana stanu bitu CIAPRB na przeciwny (logiczne NOT),
0 — sposób wyprowadzania danych: ustawienie bitu 6 CIAPRB na 1 cykl zegara,
bit 3 : 1 — jednokrotne zliczanie timera A do 0,
0 — ciągłe zliczanie od zadanej wartości.
bit 4 : 1 — rozpoczęcie zliczania od zadanej wartosci,
0 — kontynuowanie zliczania,
bit 5 : 1 — timer A pracuje w oparciu o zegar systemowy,
0 — timer A zalicza impulsy z linii CNT1 (4 złącze USER PORT).
bit 6 : 1 — port szeregowy (CIASDR) pracuje jako nadajnik,
0 — port szeregowy pracuje jako odbiornik,
bit 7 : 1 — zegar czasu rzeczywistego jest taktowany częstotliwością 50 Hz,
0 — zegar czasu rzeczywistego jest taktowany częstotliwością 50 Hz.

Właściwie pozostał mi tylko komentarz do bitów 5 i 7. Jeżeli bit 5 jest wyzerowany, to timer A zlicza impulsy z linii CNT1. Pozwala to na wykorzystanie timer A jako np. miernika częstotliwości. W przeciwnym wypadku timer A zlicza impulsy zegara systemowego, które dochodzą do niego 985250 razy na sekundę (1022730 dla NTSC). Bit 7 umożliwia wybór częstotliwości taktującej zegar czasu rzeczywistego. Częstotliwość sieci elektrycznej w Europie wynosi 50 Hz, więc ten bit powinien być ustawiony. Należy zauważyć, że częstotliwość napięcia w polskich sieciach elektrycznych nie jest zbyt regularna, co powoduje pewne niedokładności w działaniu zegara. W USA częstotliwość sieci wynosi 60 Hz — bit 7 powinien więc być wyzerowany.

56335 ($DC0F) CIACRB CIA#1+15
Rejestr kontrolny B.
bity 0–4 : znaczenie identyczne jak w CIACRA, ale dotyczy timera B. Sygnał przejscia przez 0 to 7 bit CIAPRB,
bity 5–6 : 00 — timer B zlicza cykle zegara systemowego,
01 — timer B zlicza impulsy na linii CNT1,
10 — zliczanie przejść przez 0 timera A (32-bitowy timer),
11 — zliczanie przejść przez 0 timera A, jeśli pojawi się impuls na linii CNT1,
bit 7 : 1 — zapis do rejestrów zegara czasu rzeczywistego ustawia zegar,
0 — zapis do rejestrów zegara czasu rzeczywistego ustawia „alarm"

Obszar pamięci 56336–56575 ($DC10–$DCFF)
„Lustrzane odbicie" rejestrow CIA#1. Podobnie jak w wypadku układu SID (C&A 04/92) układ CIA 1 ma zwierciadlane odbicia swoich rejestrów w pamięci. Wynika to z prostego faktu, iż układy CIA adresują 256 bajtów, a wykorzystują . . tylko 16 z nich. Słowem (a może raczej wzorem) 56320+K*16 (gdzie K=1,2,3,...,15) to ten sam rejestr. Układ CIA nr 1 charakteryzuje się (jak widać) niesamowitym upakowaniem informacji — w 16 bajtach mieści się to, co można by upchnąć w 256 bajtach. Ciekawskich i zgłodniałych zapraszam na kolejny obiadek z układem CIA #2 za miesiąc.

BARTŁOMIEJ DRAMCZYK

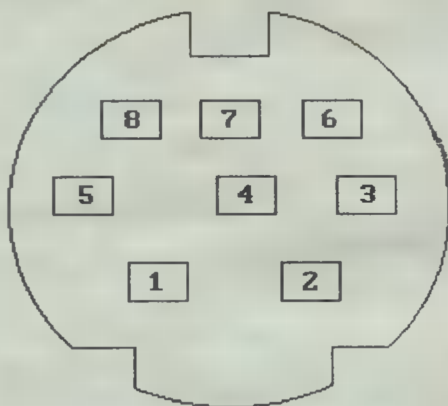
Literatura:
S. Leemon: „MAPPING THE C-64 AND C-64C"
M.L. De Jong: „Assembly Language Programming with the C-64"

KABELKOLOGIA C-16/116/PLUS/4

Firma Commodore wprowadzając na rynek komputery rodziny C16 zmieniła gniazda magnetofonu i joysticka. Co dziwniejsze, reszta pozostała bez zmian, to znaczy w dalszym ciągu można przyłączyć do C-16 magnetofon jak i joystick od C-64 — wystarczy tylko zakupić odpowiednią przejściówkę, czyli kawałek kabelka, który z jednej strony jest zakończony wtyczką pasującą do „szesnastki" a z drugiej gniazdkiem, do którego da się przyłączyć normalny joystick lub magnetofon.

Większość Czytelników ma magnetofony firmowe, standardowo wyposażone w okrągłą wtyczkę, pozostaje jednak problem joysticka. Osobom, które nie chcą kupować na warszawskiej giełdzie polecam samodzielne wykonanie takiej przejściówki.

Pierwszym problemem, który musimy w tym celu rozwiązać jest sposób „podłączenia się" do komputera. Kupno odpowiedniej wtyczki niestety nie wchodzi raczej w grę, gdyż jest ona praktycznie nie do zdobycia — pozostają więc dwa rozwiązania: wykonanie jej samodzielnie lub (mniej eleganckie) przylutowanie jednego końca naszej przejściówki bezpośrednio do gniazda (od wewnątrz) lub do płyty komputera. Ze zdobyciem drugiego końca przejściówki nie powinno być większych kłopotów — gniazda normalnego joysticka są nawet produkowane w Polsce i można je kupić w sklepach z częściami elektronicznymi. Całość łączymy korzystając z zamieszczonych obok tabel i rysunków (rysunki gniazd przedstawiają wyprowadzenia widziane z zewnątrz komputera).



Widok gniazda joysticka w C-16, C-116, PLUS/4

Posiadacze C-16 mogą umieścić standardowe gniazda joysticków na stałe — z lewej strony komputera jest na to sporo miejsca. Wymaga to wprawdzie wiercenia otworów w obudowie, lecz komputer w ten sposób zyska na wartości.

Na koniec uwaga: **NIE WOLNO PRZEŁĄCZAĆ JOYSTICKÓW PRZY WŁĄCZONYM KOMPUTERZE!** Grozi to uszkodzeniem układu TED, który kosztuje około 300 tysięcy złotych, a poza tym bardzo trudno jest go zdobyć. Objawy uszkodzenia tego układu są różne: mogą to być „śmieci" widoczne na ekranie podczas wczytywania z magnetofonu, nieprawidłowe działanie przycisku „FIRE" lub nieodczytywanie przez komputer niektórych „kierunków" joysticka. Za część tych objawów może być także odpowiedzialny uszkodzony joystick, lecz jego działanie można sprawdzić posługując się baterią i żarówką (aby ułatwić jego sprawdzanie, zamieszczam odpowiedni schemat). Do testowania komputera może się przydać podany niżej króciutki programik:

```
1PRINT JOY(1): GOTO 1:REM *** TEST PORTU 1 ***
```

lub

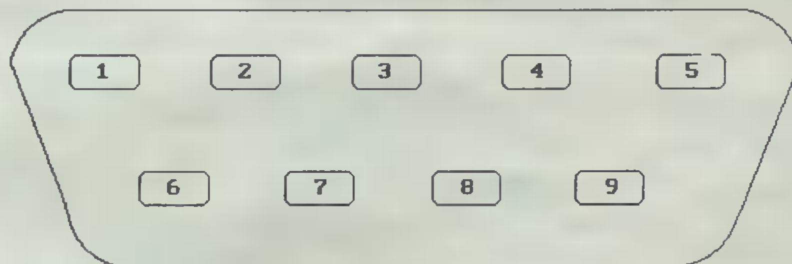
```
1 PRINT JOY(2): GOTO 1:REM *** TEST PORTU 2 ***
```

Po wpisaniu RUN powinien się pokazać rząd cyfr, na początku zer, po wciśnięciu przycisku FIRE - 128, a przy wychylaniu drążka odpowiednio według schematu:

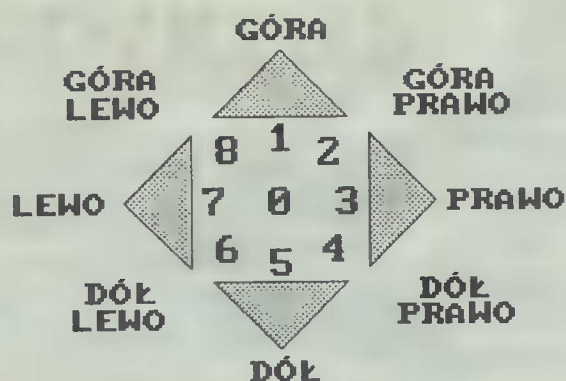
```
       góra
      8  1  2
lewo  7  0  3  prawo
      6  5  4
        dół
```

Jeżeli np. przytrzymując przycisk FIRE ciąg liczb 128 będzie czasami przerywany zerami, to może być to znak, że dni joysticka są policzone albo — że Twój TED jest uszkodzony. W każdym bądź razie po stwierdzeniu dziwnych objawów powinieneś oddać komputer do wyspecjalizowanego warsztatu.

WOJCIECH KAZIMIERCZAK



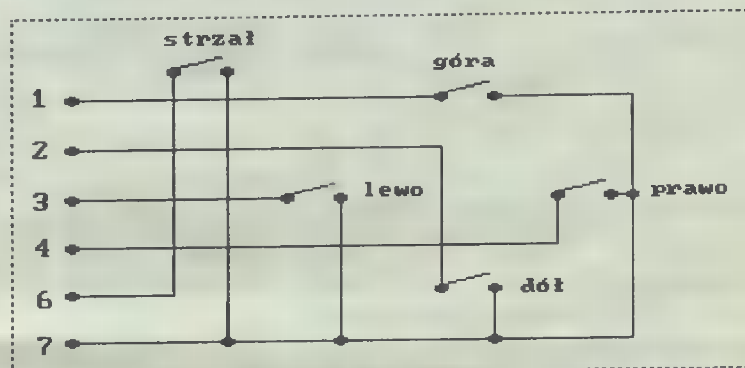
Widok gniazda joysticka w C-64



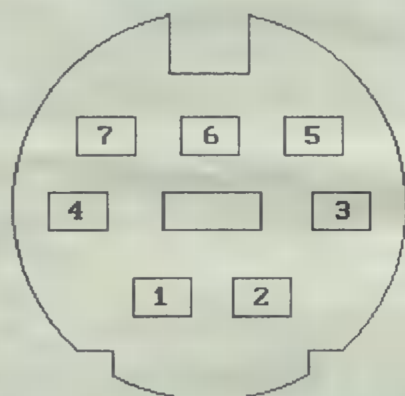
STRZAŁ +128

Wartości uzyskiwane z odczytu portu joysticków za pomocą funkcji JOY(n).

C-16, PLUS/4	Sygnał	C-64
1	styk 0	1
2	styk 1	2
3	styk 2	3
4	styk 3	4
5	+5 V	7
6	strzał	6
7	GND/masa	8
8	D1/D2	–
–	wiosełko x	5
–	wiosełko y	9

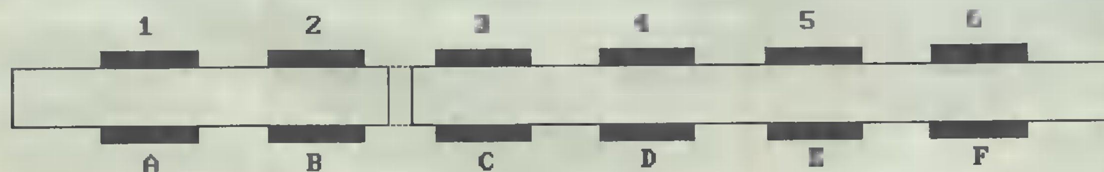


Schemat joysticka (numeracja wg oznaczeń C 16)



Widok gniazda magnetofonu w C 16, PLUS/4 (z zewnątrz)

C 16, PLUS/4	Sygnał	C 64
1	GND/masa	A-1
2	+5 V	B-2
3	silnik	C-3
4	odczyt	D-4
5	zapis	E-5
6	przełącznik	F-6
7	GND/masa	–



Widok złącza magnetofonu w C 64 (z zewnątrz)

DUSZNY KURSOR

Czy zdarzało Ci się kiedyś, że miałeś już dosyć patrzenia na mrugający wesoło prostokącik, zwany z angielska „kursorem"? Jeśli nie, to możesz ze spokojnym sumieniem darować sobie lekturę tego artykułu — nie ma tu nic dla Ciebie.

Zajmiemy się bowiem problemem likwidacji kursora i zastąpienia go czym innym, spełniającym tę samą funkcję. Najlepsze są pod tym względem duszki — można łatwo zmieniać ich położenie, nie ma także problemu ze „śladem" pozostawianym przez zwykłe znaki. Trzeba jedynie wziąć z pamięci współrzędne wskazujące położenie, w którym kursor miał się znajdować, pomnożyć je przez 8, wstawić do komórek $d000, 53248 i $d001, 53249 i... w tym miejscu zaczynają się schody.

Jak wiadomo linia ekranu ma 40 znaków, zaś linia logiczna, w której może znajdować się kursor — 80. Widać to np. podczas wpisywania programu w BASIC — jeden wiersz może zajmować dwie linie ekranowe. Z tego też powodu nie zawsze można „przeciągnąć" za pomocą klawisza „DEL" (INST/DEL) jakiś tekst do poprzedniej linii. Również z tego samego powodu komputer czasem wstawia, a czasem nie wstawia pustej linii, gdy napiszemy co w ostatniej kolumnie. Dodatkowo musimy sprawdzać, czy kursor jest w pierwszych czy ostatnich 40 znakach, i w zależności od tego te 40 odejmować. Ostatnim kłopotem jest fakt, że ekran ma w poziomie 320 punktów, a duszek może być przesuwany w obrębie 255 z nich — dalej — trzeba już zmieniać bit w komórce $d010, 53264. No, potem trzeba do pomnożonej przez 8 liczby dodać 24 ($18), by duszek nie chował się za ramką. Jeżeli dodam, że pozycja kursora w linii logicznej jest zapisana w komórce 211 ($d3), a kolumna, w której siedzi kursor — w komórce 214 ($d6), to powinno Wam to wystarczyć, aby bez większych kłopotów napisać sobie taki program samemu.

Jeżeli jednak nie jesteś szczególnie biegły w programowaniu w asemblerze, nie czujesz się na siłach, albo po prostu nie chce Ci się myśleć, musisz wpisać wydrukowany obok listing (lub kupić naszą dyskietkę nr 08/92). Przygotowałem go za pomocą programu Ign-Data Maker 2, zamieszczonego w tym numerze. Po wpisaniu, zapisz program na dyskietce (SAVE "DUSZNY KURSOR",8) lub taśmę (SAVE "DUSZNY KURSOR",1). Program uruchamia się za pomocą RUN. Kiedy komputer zamelduje, że "duszny kursor" jest gotów do pracy, wpisz SYS 52992 i naciśnij RETURN. Jeśli nie spodoba Ci się zaproponowany przeze mnie wzór kursora, zawsze możesz go sobie zmienić. Najprościej będzie to zrobić za pomocą monitora w module Final III — wpisujesz ES cf70 i duszek pojawi się na ekranie. Jeśli nie, to musisz zaprojektować go za pomocą jakiegokolwiek dostępnego programu i przenieść go tak, aby był on zapisany w pamięci od komórki 53104 ($cf70) począwszy.

BARTEK I. KACHNIARZ

```
400 read n$,po,k
405 for a=1 to 6
410 read a(a):b=b+a(a)
415 if a(a)<0 then 470
420 next
425 read c
430 if c<>b then print"blad w linii";nl
    +1000:stop
435 b=0
440 nl=nl+5
445 for a=1 to 6
450 poke po,a(a)
455 po=po+1
460 next
465 goto 405
470 print"program "n$:print"gotowy."
475 :
995 data duszny kursor, 52992 , 53168
1000 data 120,173,020,003,141,109,566
1005 data 207,173,021,003,141,110,655
1010 data 207,169,207,141,021,003,748
1015 data 169,049,141,020,003,169,551
1020 data 001,013,021,208,141,021,405
1025 data 208,169,011,141,248,007,784
1030 data 162,064,189,111,207,157,890
1035 data 191,002,202,208,247,088,938
1040 data 096,165,211,201,040,144,857
1045 data 002,233,040,162,003,010,450
1050 data 144,005,160,001,076,069,455
1055 data 207,160,000,202,208,243,1020
1060 data 140,016,208,024,105,024,517
1065 data 144,005,160,001,140,016,466
1070 data 208,141,000,208,165,214,936
1075 data 010,010,010,105,050,141,326
1080 data 001,208,173,134,002,141,659
1085 data 039,208,169,003,133,205,757
1090 data 076,049,234,000,000,000,359
1095 data 000,000,128,000,000,192,320
1100 data 000,000,224,000,000,224,448
1105 data 000,000,224,000,000,224,448
1110 data 000,000,224,000,127,096,447
1115 data 000,063,160,000,031,192,446
1120 data 000,000,000,000,000,000,000
1125 data 000,000,000,000,000,000,000
1130 data 000,000,000,000,000,000,000
1135 data 000,000,000,000,000,000,000
1140 data 000,000,000,000,000,000,000
1145 data 000,000,000,000,000,000,000
1150 data -1
1155 rem linie data: ign-datamaker
```

IGN DATAMAKER 2

Program Ign-DataMaker 2 powstał, by raz na zawsze rozwiązać problem „Jak to przenieść". Powstał też po to, by nie trzeba było przeglądać trzech kilometrów wydruku, bo komputer wyświetlił komunikat „ZLE DANE".

Nie znaczy to wcale, że cele te zostały w 100% spełnione. Ogólnie przecież wiadomo, że program jest bezbłędny, dopóki ktoś nie znajdzie w nim błędu. Wydaje mi się jednak, że gra warta jest świeczki, bo program wykazał się już w warunkach bojowych — czego dowodem jest program „Duszny kursor", opublikowany gdzies w okolicy.

Ign-DataMaker 2 różni się od poprzednich wersji kilkoma istotnymi cechami:

1) Nie ma błędów (patrz powyżej)
2) Pozwala na wyszukanie błędu z dokładnością do jednego wiersza.
3) Po uruchomieniu zostawia w pamięci wyłącznie program wynikowy, gotowy do akcji.

Przy pisaniu tego programu posłużyłem się metodą „dynamicznej klawiatury". Pozwala to na wydrukowanie na ekranie linii danych a następnie podstępne ich potwierdzenie przez wmówienie komputerowi, że ktoś nacisnął „RETURN" i to dwa razy: raz, by wbić do pamięci tę linię i drugi, by wykonać instrukcję GOTO, niezbędną dla dalszego działania programu. Dzięki tej „dynamicznej klawiaturze" komputer wpisuje nasz program w języku maszynowym, dźwięk lub grafikę w linie danych, a następnie sam się niszczy.

Ign-DataMaker 2 pozwala na dokładne lokalizowanie błędów. Wprawdzie spowalnia to jego działanie o kilkanaście procent, ale wydatnie zmniejsza ryzyko konieczności przeglądania dziesiątków (jeśli nie setek) wierszy zawierających instrukcje DATA i kolumny cyferek. Tym razem komputer nie pisze „ZLE DANE", ale „BLAD W LINII 1080", co czas ewentualnych poszukiwań skraca do (góra) kilkunastu sekund.

Program należy zapisać BEZPOŚREDNIO po wpisaniu i przed zapisem POD ŻADNYM POZOREM NIE URUCHAMIAĆ! Może to spowodować utratę owoców waszego mozolnego stukania. Zapisz program za pomocą instrukcji SAVE „IGN-DATA MAKER 2",1 — jeśli używasz magnetofonu lub SAVE „IGN-DATA MAKER 2",8 — jeśli Twoją pamięcią masową jest stacja dysków. Jeżeli chcesz któryś ze swoich programów mieć zapisany w postaci wierszy z DATA, musisz:

1) wczytać IGN-DATA MAKER 2 do pamięci
2) uruchomić go (RUN)
3) wpisać nazwę programu
4) wpisac adres komorki, od której chcesz rozpocząć przepisywanie programu
5) wpisać adres końca programu
6) poczekać kilka minut
7) zapisać swój program na taśmę lub dyskietkę.

...i to wszystko.

BARTEK I. KACHNIARZ

```
100 input"nazwa programu";n$
102 input "poczatek";p
104 input "koniec";k
106 print"999 data";n$","p","k:pr
    int"goto 114
108 poke 631,19:poke 632,13:poke
    633,13
110 poke 198,3
112 stop
114 read a$,p,k
116 for a=p to p+6
118 a$(a-p+1)=str$(peek(a))
120 a$(a-p+1)=right$(a$(a-p+1),le
    n(a$(a-p+1))-1)
122 next
124 print"";1000+ln;"data 000,000
    ,000,000,000,000,0
126 print"goto180"
128 ln=ln+5
130 print"";
132 for x=1 to 6
134 on len(a$(x))gosub200,202,204
136 sc=sc+val(a$(x))
138 next
140 sc$=str$(sc)
142 sc$=right$(sc$,len(sc$)-1)
144 print"";sc$;
146 sc=0
148 print
150 poke 631,19:poke 632,13:poke
    633,13
152 poke 198,3
154 k1=int(k/256)
156 k2=k-k1*256
158 poke 1500,k1
160 poke 1501,k2
162 p1=int(p/256)
164 p2=p-p1*256
166 poke 1502,p1
168 poke 1503,p2
170 l1=int(ln/256)
172 l2=ln-l1*256
174 poke 1504,l1
176 poke 1505,l2
178 stop
180 k=peek(1500)*256+peek(1501)
182 p=peek(1502)*256+peek(1503)
184 ln=peek(1504)*256+peek(1505)
186 p=p+6:if p<=k then 116
188 print"";1000+ln;"data -1"
190 print1005+ln;"rem linie data:
    ign-datamaker"
192 print"goto 206"
194 poke 631,19:poke 632,13:poke
    633,13
196 poke 634,13:poke 198,4
198 stop
200 print"";a$(x);:return
202 print"";a$(x);:return
204 print"";a$(x);:return
206 print"";:poke631,19
208 for a=1to 10:print2*a+98+x*20
210 poke 631+a,13:next:poke1500,x
212 poke 198,12:poke 642,13
214 print"goto216":stop
216 x=peek(1500):x=x+1
218 goto 206
```

DWA EKRANY NA JEDNYM

Szanowna Redakcjo!

Zamieszczony program umożliwia rozszerzenie możliwości graficznych C-64 o kilka użytecznych funkcji. Dzięki zastosowaniu techniki przerwań możliwe jest uzyskanie na ekranie rownocześnie dwóch obszarów różnej rozdzielczości. Wymiary tych obrazów mogą być regulowane poprzez modyfikację zawartości dwóch komórek pamięci: $908E (37006) i $9083 (36995). Określają one numery linii ekranu stanowiące początek i koniec obrazu wysokiej rozdzielczości. Można więc uzyskać w tej samej chwili np. dwa obrazy, niskiej i wysokiej rozdzielczości.

Pamięć obrazu wysokiej rozdzielczości zaczyna się od adresu $A000 (40960), tzn. umieszczona jest w miejscu interpretera BASIC. Aby umożliwić odczytywanie zawartości tej pamięci została wprowadzona odpowiednia instrukcja: **@P,n**, która w komórce pamięci o adresie 2 umieszcza zawartość komórki pamięci o adresie n. Program zapewnia możliwość pracy z dwoma obrazami wysokiej rozdzielczości. Pamięć drugiego obrazu umieszczona jest w miejscu systemu operacyjnego (od $E000). Zamianę tych dwu obrazów umożliwia instrukcja **@S**. W przypadku pracy w trybie wysokiej rozdzielczości wystąpienie błędu w programie powoduje automatyczne przełączenie obrazu na ekran tekstowy.

Pamięć koloru dla grafiki wysokiej rozdzielczości (HIRES) rozpoczyna się od adresu $8C00 (35840). Program maszynowy zajmuje obszar $9000–$9190 (36864–372642) i może być uruchamiany instrukcją **SYS 9*16^3** (np. po wyjściu do innych programów rozszerzających typu TURBO TAPE). Pomiędzy programem i pamięcią wysokiej rozdzielczości znajduje się wolna przestrzeń (ok. 3.5 kB), którą można wykorzystać np. na dołączenie dalszych instrukcji (graficznych, dźwiękowych). Program maszynowy można dołączyć, w postaci loadera, do własnego programu napisanego w BASIC, unikając w ten sposób konieczności jego każdorazowego wczytywania. Poniżej podany jest zestaw wszystkich instrukcji:

- **@H** (High Res) — włączenie obrazu wys. rozdzielczości
- **@L** (Low Res) — włączenie obrazu niskiej rozdzielczości
- **@W** (Window) — dwa obrazy różnej rozdzielczości
- **@S** (Swap) — zamiana miejscami pamięci obrazów wysokiej rozdzielczości
- **@F,n** (Fill) — wypełnienie pamięci hires wartością n (0–255) ; n=0 powoduje wyczyszczenie ekranu
- **@C,n** — ustawienie koloru tła dla HIRES (n=0–15)
- **@P,n** — odczytanie zawartości komórki pamięci o adresie n do komórki o adresie 2.

TOMASZ BUDZYŃSKI

```
0 rem *** t.budzynski ***
1 poke 55,255:poke 56,139:clr
2 ad=9*16^3:su=0
3 for i=0 to 399
4 read d:poke(ad+i),d:su=su+d
5 next
6 if su=50122 then 8
7 print "blad w danych":stop
8 sys ad:@f,0:@c,13:@w:stop
9 :
10 data 162,156,160,144,142,008,003,140
11 data 009,003,162,070,160,145,142,000
12 data 003,140,001,003,096,160,010,136
13 data 208,253,173,017,208,041,223,141
14 data 017,208,169,151,141,000,221,169
15 data 021,141,024,208,096,160,010,136
16 data 208,253,169,061,141,024,208,173
17 data 017,208,169,059,141,017,208,169
18 data 149,141,000,221,096,141,232,143
19 data 238,070,144,208,003,238,071,144
20 data 236,070,144,208,240,204,071,144
21 data 208,235,096,120,169,049,160,234
22 data 141,020,003,140,021,003,169,000
23 data 141,026,208,088,032,021,144,096
24 data 173,025,208,141,025,208,016,029
25 data 173,017,208,041,032,208,011,032
26 data 045,144,169,144,141,018,208,076
27 data 188,254,032,021,144,169,000,141
28 data 018,208,076,188,254,173,013,220
29 data 088,076,049,234,032,115,000,240
30 data 004,201,064,240,003,076,231,167
31 data 032,115,000,072,032,115,000,104
32 data 201,076,208,006,032,091,144,076
33 data 174,167,201,070,208,027,032,253
34 data 174,032,158,183,138,162,000,160
35 data 160,142,070,144,140,071,144,162
36 data 064,160,191,032,069,144,076,174
37 data 167,201,067,208,027,032,253,174
38 data 032,158,183,138,162,000,160,140
39 data 142,070,144,140,071,144,162,232
40 data 160,143,032,069,144,076,174,167
41 data 201,072,208,009,032,091,144,032
42 data 045,144,076,174,167,201,080,208
43 data 026,032,253,174,032,138,173,032
44 data 247,183,169,054,133,001,162,000
45 data 161,020,133,002,169,055,133,001
46 data 076,174,167,201,087,208,040,120
47 data 162,112,160,144,142,020,003,140
48 data 021,003,173,017,208,041,127,141
49 data 017,208,169,129,141,026,208,088
50 data 076,174,167,076,008,175,138,048
51 data 003,032,091,144,076,139,227,201
52 data 083,208,240,120,169,052,133,001
53 data 162,000,160,160,134,251,132,252
54 data 160,224,134,253,132,254,160,191
55 data 161,251,072,161,253,129,251,104
56 data 129,253,230,251,208,002,230,252
57 data 230,253,208,002,230,254,196,252
58 data 208,230,169,065,197,251,208,224
59 data 169,055,133,001,088,076,174,167
```

O PRZERWANIACH GRAFICZNYCH

Będzie to (po raz kolejny) strzelanie z dział elektronowych. Tym razem chciałbym zaprezentować trzy proste programiki wykorzystujące przerwania rastra. Zabawa jest o tyle prosta, że wszystko polega na szybkiej zmianie wartosci komórek 53280 ($DO20) i 53281 ($DO21). Z poziomu BASIC nie wygląda to zbyt zachęcająco. Coś zaczyna migać, ogólnie wygląda to na idealny program do niszczenia oczu — i to mają być te osławione przerwania rastra? Ale z poziomu asemblera zaczyna to wszystko mieć ręce i nogi. Popatrz na trzy poniższe linie programu:

```
BEGIN  INC $D020
       INC $D021
       JMP BEGIN
```

Ot właśnie. Krótkie i działa. I nareszcie widać coś ciekawszego na ekranie. Poprzez rozbudowę tego krociutkiego programiku, dopisywanie pętli opóźniających, zmianę odstępów pomiędzy zwiększaniem lub zwiększaniem zawartosci komórek 53280 i 53281 można dojść do całkiem przyzwoitych efektów nie martwiąc się nawet o budowanie tabeli opóźnień. Właśnie w taki sposób powstał program oznaczony jako RASTER 1. Właściwie nie ma o czym dalej pisać. Wystarczy go uruchomić i sprawdzić.

Ciekawszą zabawę oferuje program oznaczony jako RASTER 2. W tym programie zastosowałem jedną pętlę opóźniającą, ktorej ilość powtórzeń można regulować; druga pętla określa liczbę powtórzeń zmian pętli koloru (jej długość także można zmieniać. Po uruchomieniu programu dostępne są następujące funkcje:

C=	—	zmniejszenie ilości powtórzeń pętli koloru o 1;
RUN/STOP	—	zwiększenie ilości powtórzeń pętli koloru o 1,
CTRL	—	zmniejszenie ilości powtórzeń pętli opóźniającej o 1;
←	—	zwiększenie ilości powtórzeń pętli opóźniającej o 1;
1	—	włączenie ekranu (ekran tekstowy jest widoczny);
2	—	wyłączenie ekranu (ekran tekstowy jest niewidoczny);
spacja	—	losowe (zależne od czasu naciśnięcia klawisza) zwiększenie ilości powtórzeń pętli koloru;
Q	—	losowe (zależne od czasu nacisnięcia klawisza) zwiększenie ilości powtórzeń pętli opóźniającej.

```
200 rem *********************
210 rem *     raster 1      *
240 rem *  by voyager /bad  *
260 rem * (c)  c&a magazine *
280 rem *********************
290 :
300 for a=49152 to 49188
310 :read b
320 :poke a,b
330 :k=k+b
340 next a
350 read s
360 if s<>k then print "error!":stop
370 sys 49152
380 :
400 data 120,169,000,141,032,208,141
410 data 033,208,238,032,208,238,033
420 data 208,162,000,232,238,032,208
430 data 238,033,208,206,032,208,206
440 data 033,208,224,012,208,239,076
450 data 009,192,5213
```

Należy się tu jeszcze jedno wyjaśnienie — dlaczego taki, a nie inny układ klawiszy? Otóż C-64 odczytuje klawiaturę w dosyć specyficzny sposób (patrz artykuły omawiające układ 6526, COMPLEX INTERFACE ADAPTER). Krótko mówiąc wszystkie powyższe klawisze leżą w jednym wierszu matrycy klawiatury i spawdzenie, który z nich został naciśnięty, polega tylko na zadaniu tego wiersza do testowania (LDA #$7F, STA $DC00) i sprawdzeniu, czy jakiś bit w komórce $DC01 został wyzerowany. Potem wystarczy tylko wskazać co komputer ma zrobić i gotowe. To właściwie wszystko co do programu 2. Jak działa — sprawdźcie sami.

```
200 rem ****************************
220 rem *        raster demo       *
240 rem *     by voyager /bad      *
260 rem *  (c) 1992   c&a magazine *
280 rem ****************************
290 :
300 for a=49152 to 49311
310 :read b
320 :poke a,b
330 :k=k+b
340 next a
350 read s:if s<>k then print"error
    in data!!":end
360 sys 49152
370 :
400 data 169,000,141,032,208,141,033
410 data 208,170,232,168,200,234,234
420 data 234,234,192,000,208,247,234
430 data 234,234,234,234,234,234,234
440 data 234,234,234,234,238,032,208
450 data 238,033,208,169,255,141,002
460 data 220,169,000,141,003,220,169
470 data 127,141,000,220,173,001,220
480 data 201,239,240,042,201,191,240
490 data 044,201,127,240,046,201,223
500 data 240,048,201,253,240,050,201
510 data 251,240,052,201,254,240,054
520 data 201,247,240,061,173,001,220
530 data 201,255,208,249,224,043,208
540 data 166,076,000,192,238,096,192
550 dats 076,095,192,238,017,192,076
560 data 095,192,238,096,192,076,088
570 data 192,206,096,192,076,088,192
580 data 238,017,192,076,088,192,206
590 data 017,192,076,088,192,173,017
600 data 208,009,016,141,017,208,076
610 data 000,192,173,017,208,041,239
620 data 141,017,208,076,000,192
621 data 24721
```

Ostatni program powstał w wyniku zabawy z programikiem PRZERWANIA GRAFICZNE (dodatek do „Bajtka" z 1987 roku). Program ten wyświetlał jedną linię rastra po środku ekranu wyświetlając na całym ekranie polską flagę. Wpadłem na pomysł, żeby linię rastra wprawić w ruch. Potem dołączyłem do tego jeszcze jedną, a cały efekt można zobaczyć uruchamiając program oznaczony jako RASTER D5MO 3. Program wykorzystuje przerwania IRQ. stąd też nie przeszkadza mu na przykład jednoczesna praca z interpreterem BASIC, no ale jednoczesne wykorzystanie przez inny program przerwań IRQ może już nastręczyć użytkownikowi pewnych problemów.

BARTŁOMIEJ DRAMCZYK

```
200 rem ********************
220 rem *     raster 2     *
240 rem *  by voyager /bad *
260 rem * (c) c&a magazine *
280 rem ********************
290 :
300 for a=49152 to 49254
310 :read b
320 :poke a,b
330 :k=k+b
340 next a
350 read s:if s<>k then print"error
    in data!!!":end
360 sys 49152
370 :
375 rem ***  kod maszynowy  ***
376 :
380 data 120,162,023,160,192,142,020
390 data 003,140,021,003,169,127,141
400 data 013,220,169,001,141,026,208
410 data 088,096,162,255,142,101,192
420 data 174,101,192,142,018,208,232
430 data 142,101,192,169,010,162,055
440 data 160,192,032,077,192,076,129
450 data 234,162,255,142,102,192,174
460 data 102,192,142,018,208,202,142
470 data 102,192,169,000,162,028,160
480 data 192,032,077,192,076,049,234
490 data 141,032,208,142,020,003,140
500 data 021,003,173,017,208,041,127
510 data 141,017,208,173,025,208,141
520 data 025,208,096,000,000,12443
```

ZABAWY z RAMKĄ

Aby uniknąć zniekształceń obrazu na brzegach monitora/telewizora oraz aby na wszystkich monitorach cały ekran roboczy był dobrze widoczny, konstruktorzy Commodore 64 wprowadzili pewne ograniczenia. Polegają one na tym, że do wyświetlania danych przeznaczono tylko pewien obszar leżący mniej więcej w środku ekranu, zwany dalej polem graficznym. Reszta obrazu to tak zwana ramka. Dzisiaj zajmiemy się problemem jak wyświetlić coś „nad" lub „pod" obszarem graficznym. Nie jest to wcale takie trudne.

Zacznijmy jednak od początku. Jak zapewne wiecie komórka $d011, a dokładniej jej trzeci bit, odpowiada za zwężenie pola graficznego w pionie. Spróbujcie wpisać:

POKE 53265,19 (spowoduje to zwężenie ekranu)
POKE 53265,27 (to natomiast przywróci stan początkowy)

Co ma to jednak wspólnego z wyświetlaniem na ramce? Otóż zawartość komórki $d011 (dziesiętnie 53265) mówi układowi VIC, ni mniej ni więcej, tylko w której linii ekranu ma zostać zakończone jego wyświetlanie i kiedy należy rozpocząć wyświetlanie ramki. Spróbujmy więc oszukać komputer: na początek każ mu wyświetlać normalny ekran (do rejestru $d011 wpisujemy wartość $1b). Układ wizyjny cały czas porównuje numer aktualnie wyświetlanej linii z numerem, w którym ma nastąpić „zamknięcie" ekranu. W tej chwili czeka on na wartość $fc. W czasie wyświetlania linii o numerze $f9 wpisz mu do $d011 wartość #$13. Nasz C-64 będzie oczekiwał, iż zamknięcie ekranu ma nastąpić w linii $f8. Ale czekać może długo, bo przecież tę linię wyświetlał już wcześniej...

W ten sposób komputer "zgubił" moment, w którym ma zostać "zamknięta" ramka. Ach, byłbym zapomniał! Aby operację tę można było powtórzyć podczas następnego odświeżania obrazu, to w linii o numerze większym od $fc (np. $ff) należy ponownie wpisać do rejestru $d011 wartość $1b.

Powiesz, że ramka jest otwarta, ale "pisać" po niej nie można. To fakt, pisanie tekstu albo wyświetlanie grafiki nie jest możliwe w tym miejscu. Możesz więc sobie pomyśleć: po co więc otwierać w takim razie ramkę?!? Co prawda, normalnej grafiki nie możemy tam wyświetlić, ale za to duszki na ramce wyświetla się tak samo łatwo jak na zwykłym ekranie. Przykład takiego efektu zobaczysz po wpisaniu programu 1.

Jednak duszki to nie jest jedyny sposób, w jaki można pokazać coś na dolnej lub górnej ramce. Jeśli wpisałeś już program z listingu 1 i uruchomiłeś go, to spróbuj teraz wpisywać różne wartości do komórki $3fff. Zauważysz z łatwością czarne paski powstające na dolnej i górnej ramce (niestety zmiana ich koloru, o ile mi wiadomo, nie jest możliwa). Wygląda to tak, jakbyś wyświetlał ekran wysokiej rozdzielczości i całą pamięć ekranu zapisał tą samą wartością. Wykorzystując ten fakt możesz wywołać ciekawy efekt. Jeśli w każdej następnej linii będziesz zmieniał w odpowiedni sposób komórkę $3fff to możesz spowodować, że utworzony zostanie pionowy napis powtórzony 40 razy obok siebie. Jeżeli nie rozumiesz o co mi chodzi to spróbuj wpisać i przeanalizować program z listingu 2.

Ciekawe efekty można uzyskać umieszczając na ramce rozszerzone duszki i używając efektu $3fff, ale jest to dosyć skomplikowane więc nie będziemy się tym dzisiaj zajmować.

Pamiętać należy, że w wypadku zmiany banku układu VIC zamiast korzystać z komórki $3fff korzystamy z ostatniej komórki tego banku.

RAFAŁ PIASEK

PROGRAM 1

```
100 rem *************************
110 rem *coded by jetboy/parados*
120 rem *************************
130 d=52224:b= 25519
140 c=0:e=d
150 read a$:if a$="end" then 250
160 a1=asc(left$(a$,1))and63
170 a2=asc(right$(a$,1))and63
180 if a1>47 then 200
190 a1=a1+9:goto 210
200 a1=a1-48
210 if a2>47 then a2=a2-48:goto 2
    30
220 a2=a2+9
230 a=a1*16+a2:poke d,a
240 d=d+1:c=c+a:goto 150
250 if c<>b then print "blad w li
    niach data":stop
260 sys e
270 data 78,20,5b,ff,a2,cc,a0,77
280 data 8c,14,03,8e,15,03,a2,1b
290 data 8e,11,d0,a2,f1,8e,1a,d0
300 data a2,7f,8e,0d,dc,a2,00,8e
310 data 0e,dc,a2,f9,8e,12,d0,a2
320 data 00,8e,ff,3f,a2,ff,8e,15
330 data d0,8e,1b,d0,a2,00,bd,90
340 data cc,9d,c0,3f,e8,e0,42,d0
350 data f5,a2,00,a9,00,9d,27,d0
360 data e8,e0,08,d0,f8,a2,00,a9
370 data 48,9d,00,d0,69,1a,e8,e8
380 data e0,10,d0,f5,a2,00,a9,ff
390 data 9d,01,d0,e8,e8,e0,10,d0
400 data f7,a2,00,a9,ff,9d,f8,07
410 data e8,e0,08,d0,f8,58,60,a9
420 data 01,8d,19,d0,a2,13,8e,11
430 data d0,a2,00,ec,12,d0,d0,fb
440 data a2,1b,8e,11,d0,4c,31,ea
450 data 00,00,00,00,00,00,00,00
460 data 00,07,80,3f,1f,80,7f,3f
470 data 80,7f,78,00,f7,70,40,e7
480 data e0,a1,e7,e0,c1,c7,e1,23
490 data c7,e0,d3,ff,e0,07,ff,70
500 data 07,ff,78,0f,07,3f,ff,07
510 data 1f,fe,07,07,fe,07,00,00
520 data 00,00,00,00,00,00,00,00
530 data 00,00,end
```

PROGRAM 2

```
100 rem *************************
110 rem *coded by jetboy/parados*
120 rem *************************
130 d=49152:b= 18364
140 c=0:e=d
150 read a$:if a$="end" then 250
160 a1=asc(left$(a$,1))and63
170 a2=asc(right$(a$,1))and63
180 if a1>47 then 200
190 a1=a1+9:goto 210
200 a1=a1-48
210 if a2>47 then a2=a2-48:goto 2
    30
220 a2=a2+9
230 a=a1*16+a2:poke d,a
240 d=d+1:c=c+a:goto 150
250 if c<>b then print "blad w li
    niach data":stop
260 sys e
270 data 78,20,5b,ff,a2,c0,a0,36
280 data 8c,14,03,8e,15,03,a2,1b
290 data 8e,11,d0,a2,f1,8e,1a,d0
300 data a2,7f,8e,0d,dc,a2,00,8e
310 data 0e,dc,a2,f9,8e,12,d0,a2
320 data 00,8e,ff,3f,a2,06,8e,20
330 data d0,8e,21,d0,58,60,a9,01
340 data 8d,19,d0,a2,13,8e,11,d0
350 data a2,00,ec,12,d0,d0,fb,a2
360 data 00,bd,64,c0,ac,12,d0,cc
370 data 12,d0,f0,fb,8d,ff,3f,e8
380 data e0,2a,d0,ed,a2,1b,8e,11
390 data d0,4c,31,ea,00,7c,e6,e0
400 data e6,7c,00,00,00,7c,e6,e6
410 data fe,e6,00,e6,f6,fe,ee,e6
420 data 00,fc,e6,e6,e6,fc,00,00
430 data 00,7c,e6,e6,fe,e6,00,00
440 data 00,00,00,00,00,00,end
```

KOLIZJE, WYPADKI ZDERZENIA

Poniższy program stanowi uzupełnienie cyklu artykułów Bartka Kachniarza o układzie VIC i ma za zadanie uświadomić początkującym użytkownikom C-64 co się dzieje w komórkach 53278 i 53279 podczas zetknięcia się na ekranie dwóch lub więcej duszków lub duszka i tła. Takie zdarzenie nazywane jest potocznie „kolizją".

Jak już zapewne wiecie, wspomniane komórki pełnią rolę detektorów kolizji. W momencie, gdy następuje zetknięcie ze sobą dwóch lub więcej duszków, układ VIC generuje przerwanie i automatycznie wpisuje do komórki 53278 odpowiednią wartość. Nasuwa się pytanie — jaka to wartość, tzn. skąd wiadomo, które duszki się zderzyły?

C-64 może normalnie (czyli bez sztuczek) wyświetlić na ekranie maksymalnie 8 duszków jednocześnie. Każdy duszek ma przypisany numer, od 0 do 7. Jednak dla komputera, który „rozumie" liczby tylko w układzie dwójkowym, byłoby bardzo niewygodnie wpisywać do komórki numery duszków przedstawione w układzie dziesiętnym. Dlatego przyjęto, że każdy duszek (tzn. jego numer) będzie reprezentowany jednym bitem w bajcie. Zobaczcie, jak to wygląda:

KOMÓRKA 53278 (53279)

dwójkowo:	0	0	0	0	0	0	1	1	(= 3)
dziesiętnie:	128	64	32	16	8	4	2	1	
numer duszka:	7	6	5	4	3	2	1	0	

Oczywiście wiecie, że bit może przyjąć tylko wartość 0 albo 1. Normalnie, gdy kolizja nie wystąpiła, wszystkie bity są wyłączone, czyli mają zerową wartość. Tym samym zawartość komórki 53278 (53279) będzie równa zeru. Na powyższym rysunku włączone są jednak bity 0 i 1. Oznacza to, że zderzyły się duszki nr 0 i 1. Oczywiście z poziomu języka BASIC, wykonując PRINT PEEK (53278), otrzymamy wynik w układzie dziesiętnym, czyli 3. Aby dobrze orientować się w temacie, radzę Wam więc nauczyć się konwersji liczb z układu dziesiętnego na dwójkowy i odwrotnie. Nie jest to wcale takie trudne i myślę, że opanujecie ten problem w parę godzin.

Ustaliliśmy zatem: gdy następuje kolizja, VIC włącza poszczególne bity w odpowiedniej komórce. Załóżmy, że nastąpiła kolizja wszystkich duszków. Jaką wartość odczytamy z komórki 53278? Naturalnie 255. Jak nie wierzycie, dodajcie do siebie wszystkie dziesiętne liczby z rysunku — musi wyjść 255. Przy okazji dowiedzieliście się, że jeden bajt może przechować liczbę o maksymalnej wartości 255. Informacja ta przyda się Wam w przyszłości.

Przejdźmy teraz do omówienia programu. Po jego wpisaniu i uruchomieniu ujrzycie na ekranie dwa jadące naprzeciw siebie czołgi oraz małą przeszkodę — powiedzmy, że jest to mina. Gdy czołgi zetkną się ze sobą lub z miną, pod napisami zobaczycie, jak zmienia się wartość komórek 53278 i 53279. Działanie programu możecie oczywiście w każdej chwili przerwać naciskając klawisz STOP.

Linie 100—140 określają wygląd naszych duszków. Linia 150 czyści ekran i przyporządkowuje literce V liczbę 53248, będącą adresem początkowym układu VIC. W ten sposób nie muszę w dalszej części programu wpisywać za każdym razem pełnego adresu, np. 53278, a tylko V+numer rejestru, czyli w tym wypadku V+30.

W linii 180, poprzez wpisanie do komórek 2040 i 2041 wartości 13, ustalony zostaje kształt obu naszych duszków. Linie 185 i 190 określają pozycję początkową duszków, a w wierszu 195 duszki zostają włączone (POKE V+21,3).

Następne linie odpowiedzialne są za ruch duszków tam i z powrotem oraz za wyświetlanie zawartości komórek 53278 i 53279. Ruch duszków można przyspieszyć likwidując linię 225 — umieszczona jest w niej pętla opóźniająca.

Sądzę, że kiedy już zapoznacie się dokładnie z programem i dokonacie na nim licznych eksperymentów (na początek radzę Wam narysować dodatkową minę, dodając w linii 195 wyrażenie np.: POKE 1400, 120), większość rejestrów VIC oraz układ dwójkowy nie będą miały przed Wami żadnych tajemnic.

CGA

```
100 for j=832 to 895:read q:poke
    j,q:next
105 data 000,000,000,000,000,000,
    000,000
110 data 000,000,000,000,000,000,
    000,000
115 data 000,000,000,000,000,000,
    000,000
120 data 000,000,000,000,000,000,
    000,000
125 data 000,000,000,000,000,000,
    000,000
130 data 126,000,001,255,255,001,
    255,128
135 data 063,255,248,081,017,020,
    170,170
140 data 170,081,017,020,063,255,
    248,255
145 :
150 print chr$(147):v=53248:k=2
155 poke v+32,0:poke v+33,0
160 for i=1 to 7:print chr$(17):n
    ext
165 printtab(6)"kolizja"tab(22)"k
    olizja"
170 printtab(6)"duszek-tlo"tab(22
    )"duszek-duszek":print
175 printtab(6)"53279:"tab(22)"53
    278:"
180 poke 2040,13:poke 2041,13
185 poke v,26:poke v+2,250
190 poke v+1,102:poke v+3,108
195 poke 1390,120:poke v+21,3
200 x1=peek(v):x2=peek(v+2)
205 if x1=252 or x1<26 then k=-k
210 poke v,x1+k:poke v+2,x2-k
215 poke 1757,peek(v+31)+48
220 poke 1773,peek(v+30)+48
225 for t=1 to 50:next
230 goto 200
```

Errata do programów „Tangens" i „Antyreset #1" z numeru C&A 05/92



I znowu redaktor Chochlik zebrał swoje żniwo. Tym razem błędy wkradły się do programów drukowanych w C&A nr 05/92 (na dyskietce znajdowały się dobre wersje). W listingu programu "Tangens" (str. 26) zostały omyłkowo połknięte wszystkie „π", i tak:

jest: 140 PRINT CHR$(147):Z=/180
a powinno być: **140 PRINT CHR$(147):Z=π/180**

jest: 165 F=I/5:F$=STR$(F)+"*/36"
a powinno być: **165 F=I/5:F$=STR$(F)+"*π/36"**

jest: 310 TEXT 60,90,"—/2",1,1,8
a powinno być: **310 TEXT 60,90,"—π/2",1,1,8**

jest: 315 TEXT 241,90,"/2",1,1,8
a powinno być: **315 TEXT 241,90,"π/2",1,1,8**

W programie "Antyreset #1" (str. 22), w linii 205 ostatnia wartość powinna wynosić 194 (a nie 294).

Za powyższe błędy gorąco przepraszamy naszych Czytelników oraz autorów programów, w których znalazły się błędy. Dziękujemy również panu Mariuszowi Lisowskiemu za szybką reakcję.

MUZYKA I C-128

Ogromna popularność edytorów dźwięku jest przyczyną pewnej bezwładności myślowej u użytkowników C-64 czy C-128. Sięgają oni po te programy często bez zastanowienia, jakby nie istniała na świecie żadna inna metoda tworzenia muzyki. A przecież na C-128 można komponować naprawdę skomplikowane utwory bez „wspomagania oprogramowaniem" — w zupełności wystarczy do tego BASIC V7.0...

C-128 i C-64 mają — sprzętowo — dokładnie takie same możliwości muzyczne, gdyż wyposażono je w identyczne procesory dźwiękowe — 6581 SID (Sound Interface Device). Wiadomo jednak, że same tylko możliwości, choćby nie wiem jak wspaniałe, są niczym, jeśli nie można z nich w łatwy sposób skorzystać. I tu właśnie uwidacznia się zdecydowana wyższość C-128. Kto próbował komponować na C-64 za pomocą BASIC V2.0 wie, co to za mordęga. Język ten nie dysponuje żadnym, ściśle muzycznym poleceniem, w związku z czym wszelkie zmiany zawartości rejestrów SID musimy wprowadzać za pomocą „nieśmiertelnej" instrukcji POKE. Jesteśmy na nią zdani np. przy określaniu częstotliwości danego dźwięku, reprezentowanej — jak na ironię — nie jednym, lecz dwoma bajtami, chociaż przy siedmiooktawowej skali SID wystarczyłaby w zupełności jednobajtowa reprezentacja odpowiednio nastrojonych „fabrycznie" brzmień. Ponadto poszczególne dźwięki, włączane oczywiście instrukcją POKE, wymagają każdorazowo wyłączenia — znów bez POKE się nie obejdzie. Jakby tego nie było dość, czas trwania nut regulować można tylko prymitywnymi pętlami opóźniającymi typu FOR...NEXT — w przypadku, gdy każdy głos ma przypisany inny rytm, pętle takie stają się tylko zawadą a programowanie zamienia się w harówkę. Możemy więc śmiało stwierdzić, że ubożuchny BASIC V2.0 zupełnie nie nadaje się do tworzenia muzyki o stopniu komplikacji tylko trochę większym o gamy C-dur, nie mówiąc już o profesjonalnych, trzygłosowych kompozycjach.

Na szczęście firma Commodore, świadoma wszystkich wymienionych wyżej wad C-64, wyposażyła jego następcę — C-128 — w BASIC V7.0. Nowa, zmodernizowana wersja poszerzona została m.in. o sześć zupełnie nowych, czysto muzycznych poleceń, umożliwiających dokonywanie bezpośrednich operacji w rejestrach SID. Całą „czarną robotę", jak np. automatyczne wyłączanie dźwięku, automatyczne dostosowywanie faz dźwięku (ADSR) do tempa itp., przejmuje na siebie interpreter, dzięki czemu programowanie muzyki staje się łatwe i przyjemne. Przyszły programista-kompozytor powinien jedynie poznać funkcje poszczególnych poleceń i ich parametrów — wtedy jego poczynania, wolne od niedociągnięć warsztatowych, zasłużą, być może, na miano artystycznych... Zapoznajmy się więc z instrukcjami muzycznymi, które oferuje nam BASIC V7.0.

PLAY

Jest to najważniejsza instrukcja muzyczna C-128. Ma ona tyle parametrów i tak bogate możliwości, że tylko w oparciu o nią można by tworzyć wyrafinowaną muzykę. Oto format tej instrukcji:

PLAY „Vn On Tn Un Xn nuty"

I znaczenie poszczegolnych parametrów:

Vn — określa numer głosu (generatora) (n=1-3)

On — numer (wysokość) oktawy (n=0-6); n=4 odpowiada oktawie razkreślnej, tej pośrodku klawiatury fortepianu.

Tn — numer (rodzaj) obwiedni (n=0-9). W pamięci ROM znajduje się 10 rodzajów gotowych obwiedni. Można z nich korzystać zmieniając jedynie cyfrę w omawianym parametrze. Każdej obwiedni odpowiada inne brzmienie:
0 — fortepian
1 — akordeon
2 — organki
3 — perkusja
4 — flet
5 — gitara
6 — klawesyn
7 — organy
8 — trąbka
9 — ksylofon
O obwiedniach powiem więcej przy omawianiu ENVELOPE.

Un — siła głosu (n=0-9) dla wszystkich głosów, gdzie n=9 oznacza maksymalną głośność (zobacz też omówienie instrukcji VOL).

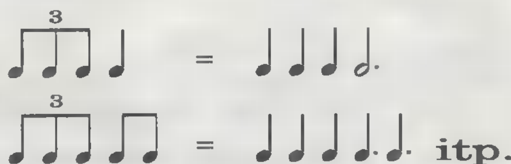
Xn — włącza (n=1) lub wyłącza (n=0) filtrowanie (patrz opis instrukcji FILTER).

Jeżeli w programie nie zdefiniujemy powyższych parametrów, komputer automatycznie przyjmuje następujące ich wartości: V1, T0, 04, U9 i X0, natomiast jako wartość rytmiczną — ćwierćnutę. Warto tu jeszcze dodać, że o ile parametr Tn dotyczy tylko jednego głosu, zdefiniowanego wyrażeniem Vn i poprzedzającego ten parametr, to parametry On, Un, Xn oraz oznaczenia rytmiczne dotyczą dźwięków bezpośrednio po nich następujących (obojętnie, w którym głosie). Co to oznacza w praktyce? To, że w utworach wielogłosowych, przed uaktywnieniem każdego głosu, musimy określić często kilka parametrów, bo zmiana jakiegoś w jednym głosie przenosi się automatycznie na inne głosy. W większości przypadków stanowi to pewną niedogodność — lepiej, gdyby poszczególne głosy „pamiętały" dane określające je, a zmiany dotyczyły tylko głosu deklarowanego. Czasem jednak utrzymanie ważności danego parametru dla następujących po nim danych może okazać się korzystne. Np. zamiast pisać PLAY „ICIDIEIFIGIAIB" wystarczy po prostu PLAY „ICDEFGAB".

nuty — **A,B,C,D,E,F,G** — nazwy dźwięków. Zamiast polskiego H występuje B, gdyż w krajach anglosaskich tak właśnie oznacza się ten dźwięk. „Nasze" B to $ z bemolem.
— krzyżyk (podwyższenie o półton)
$ — bemol (obniżenie o półton)
W — cała nuta
H — półnuta
Q — ćwierćnuta
I — osemka
S — szesnastka
. — kropka (przedłuża czas trwania dźwięku o połowę)
R — pauza; czas trwania pauzy można regulować za pomocą oznaczeń rytmu, włącznie z kropką.
M — „czekaj do końca taktu" albo „czekaj na pozostałe głosy"; służy głównie do synchronizacji głosów w czasie oraz do tworzenia przedtaktów.

Powyższe znaki nutowe, z wyjątkiem R i M, muszą zawsze poprzedzać nazwę dźwięku, np. szesnastkę *gis* trzeba zapisać jako S#G.

W oznaczeniach rytmu nie występuje, niestety, podział trójkowy. Jeśli wystąpi potrzeba użycia trioli, należy stworzyć jej ekwiwalent, wykorzystując w tym celu kropki oraz modyfi-



Rys. 1.

kując pozostałe wartości rytmiczne i tempo. Przykład przedstawiono na rysunku 1.

Spróbuj teraz zagrać na C-128 gamę C-dur. Wpisz:
PLAY „CDEFGABO5C" i naciśnij RETURN. Proste, prawda? Aby uzyskać ten sam efekt na C-64, trzeba by użyć całą masę instrukcji POKE!

W instrukcji PLAY możemy używać uprzednio zdefiniowane zmienne łańcuchowe, funkcje łańcuchowe (LEFT$, MID$, RIGHT$, STR$), lub jedno i drugie, np.:
A$=„QCEGO5C": PLAY A$ albo
PLAY A$+LEFT$(B$,3)+MID$(C$,3,7)+D$ itp.
Upraszcza to znacznie nasze programy, pozwala skrócić program i zaoszczędzić pamięć komputera, czasem przyczynia się też do uzyskania ciekawych efektów (normalnie funkcje łańcuchowe wykorzystywane są do celów zupełnie nie związanych z muzyką).

Stosowanie polecenia PLAY wymaga przestrzegania pewnych reguł, w przeciwnym razie użytkownik narażony jest na niemiłe niespodzianki. Objawiają się one w utworach dwu- lub trzygłosowych i polegają na powolnym, stopniowym „rozjeżdżaniu się" głosów (fachowo określa się to jako „wypadnięcie z taktu"). Polega to na tym, że dźwięki, które według nut powinny zabrzmieć jednocześnie (np. na raz, czyli na początek taktu), w rzeczywistości słyszymy jako arpeggio (rozłożony akord) lub — co gorsza — jako w ogóle niezsynchronizowane brzmienie. Najlepiej wyjaśnić to na przykładzie:

10 PLAY „V2O3HCV1O4SCDEFGFED"
20 PLAY „V2O3HEV1O4SCDEFGFED"
30 PLAY „V2O3HGV1O4SCDEFGFED"
40 PLAY „V2O3HEV1O4SCDEFGFED"
50 GOTO 10

Programik ma za zadanie odegrać prostą, dwugłosową melodyjkę. Uważny słuchacz zorientuje się natychmiast, że głosy nie zgadzają się rytmicznie. Gdyby program był bardziej skomplikowany i wykorzystywał trzy generatory już po paru taktach głosy rozjechałyby się o szesnastkę lub więcej. Jaka jest tego przyczyna? Otóż interpreter BASIC potrzebuje trochę czasu na odczytanie i prawidłowe przetłumaczenie na język maszynowy danych zawartych w poleceniu. Wykonanie dwóch półnut (albo czterech ćwierćnut) zajmuje zatem więcej czasu, niż jednej całej nuty, ponieważ komputer musi odczytać i wykonać dane dla dwóch (czterech) nut.

Aby uniknąć niespójności głosów przy posługiwaniu się instrukcją PLAY, trzeba stosować się do następujących zasad:

1. Głosy, które mają zabrzmieć jednocześnie, powinny znaleźć się w programie obok siebie, przy czym jako pierwszy należy wpisywać ten głos, któremu przypisane są drobniejsze wartości rytmiczne.
2. Głos kończący dany odcinek taktu (lub zmienną przypisaną instrukcji PLAY) powinien rozpoczynać następną miarę w takcie (następną zmienną).
3. Zmienna z parametrami może reprezentować dowolną, lecz pełną miarę w takcie. Np. jeśli chcemy mieć w jednym głosie ćwierćnutę, a w drugim cztery szesnastki, to zmienna powinna kończyć się zdefiniowaniem ostatniej, czwartej szesnastki (a nie, powiedzmy, trzeciej) — takie rozpisywanie melodii wprowadzi do programów porządek i ułatwi, zwłaszcza w długich kompozycjach, znalezienie wybranego taktu, bądź fragmentu melodii.

Ogólnie, podczas rozpisywania rytmu, należy być raczej ostrożnym oraz pamiętać, że:
— jeżeli dany głos zaczął już „grać" jedną nutę, druga nuta zabrzmi w tym głosie dopiero po zakończeniu pierwszej;
— jeżeli głos nie odtwarza w danej chwili dźwięku, wykona następną nutę natychmiast, niezależnie od tego, co dzieje się w innych głosach.

Uwzględniając powyższe wskazówki zmodyfikujmy nieco nasz program:
10 PLAY "V1O4SCV2O3HCV1O4SDEFGFED"
20 PLAY "V1O4SCV2O3HEV1O4SDEFGFED"
30 PLAY "V1O4SCV2O3HGV1O4SDEFGFED"
40 PLAY "V1O4SCV2O3HEV1O4SDEFGFED"
50 GOTO 10
Teraz brzmi już dobrze, prawda? Inna rzecz, że program się nieco wydłużył, ale jest na to rada: skorzystajmy ze zmiennych łańcuchowych. Zamiast wpisywać aż cztery niewiele różniące się linie, wydziel z nich identyczne elementy — reszta to tylko kwestia techniki:
10 A$="V1O4SCV2O3H":B$="V1O4SDEFGFED"
20 PLAY A$+"C"+B$+A$+"E"+B$
30 PLAY A$+"G"+B$+A$+"E"+B$
40 GOTO 20
W końcu otrzymaliśmy krótki i przejrzysty program — jest on potwierdzeniem tezy, że używanie funkcji (zmiennych) łańcuchowych w programach muzycznych znacznie ułatwia pracę. Oto inny przykład ich zastosowania:
10 A$="ICDEFGAB"
20 FOR W=1 TO 5
30 W$=STR$(W)
40 PLAY "O"+W$+A$
50 NEXT:PLAY "O6C"
Funkcja STR$ powoduje tu zmianę oktawy na coraz wyższą. Nietrudno zauważyć, że może ona być użyta do manipulacji innymi parametrami przy minimalnym zwiększeniu objętości programu.

Na zakończenie omawiania instrukcji PLAY należy się Wam jeszcze jakiś mały efekcik. Dodajcie do trzeciego przykładu linię:
5 PLAY „V1T3V2T2"
Taka muzyka mogłaby służyć za podkład do jakiegoś filmu, prawda? Użyte w taki sposób polecenia PLAY nic „nie gra", lecz ustala tylko obwiednię nr 3 dla głosu 1 (perkusja) i nr 2 (organki) dla głosu 2. Czy dostrzegacie teraz ile możliwości kryje w sobie instrukcja PLAY?

TEMPO

Za pomocą tego polecenia możemy dowolnie ustalać tempo naszej kompozycji. Format:
TEMPO n
gdzie n może przyjmować wartości od 1 (tempo najwolniejsze) do 255 (tempo najszybsze).

Tempo (wartość parametru n) możemy określać albo w drodze eksperymentów, albo korzystając z następującego wzoru:

liczba ćwierćnut na minutę = 12.5 * n

Zatem przy n=4 będziemy mieli 50 ćwierćnut na minutę (tempo bardzo wolne), a przy n=20 — 250 ćwierćnut na minutę (tempo bardzo szybkie). Jeśli w programie nie zadeklarujemy tempa, C-128 automatycznie ustawi wartość parametru n na 8.

Przypominam, że instrukcji TEMPO można używać niekoniecznie tylko na początku programu (jak to przeważnie bywa). Stosując ją w trakcie jakiegoś utworu, możemy uzyskać efekt stopniowego przyspieszenia lub zwalniania, możemy też bardziej swobodnie operować rytmem (mam tu na myśli nieregularne podziały, np. na piątki; wymaga to jednak pewnej wprawy i nie zawsze daje się zrealizować).

VOL

Format: **VOL n**, gdzie n może przyjmować wartości od 0 do 15. Polecenie to służy do ustalania siły głosu dla wszystkich trzech generatorów. Głośność można też regulować parametrem Un w instrukcji PLAY. Osobiście wolę jednak używać w tym celu VOL, ponieważ instrukcja PLAY ma już i tak bardzo dużo parametrów i lepiej jej dodatkowo nie komplikować.

VOL można stosować, podobnie jak TEMPO, w trakcie programu, co pozwala na uzyskanie efektów dynamicznych (ściszanie lub zgłaśnianie). Ilustruje to poniższy program:
10 X=2:P=2
15 VOL X:TEMPO 4+X
20 PLAY „T3O3SCEGE"
25 X=X+P
30 IF X=14 OR X=2 THEN P=-P:GOTO 15
35 GOTO 15

W następnym odcinku omówię pozostałe trzy instrukcje muzyczne BASIC V7.0. A teraz proponuję Wam wpisanie poniższego programu — wykonuje on fragment bardzo znanej uwertury do opery „Wilhelm Tell" G. Rossiniego. Spróbujcie poeksperymentować: zmieńcie parametry Tn dla każdego głosu, ustalcie inne wartości tempa i siły głosu; jeżeli skasujecie linię 100 i dopiszecie:
249 VOL 3
259 VOL 5
269 VOL 10
279 VOL 15
290 GOTO 249
uzyskacie dodatkowy efekt narastającej dynamiki.

CHRISTIAN GRZENKOWICZ

```
100 VOL 15
110 TEMPO 16
120 PLAY"V1T5V2T4V3T4"
130 A$="V1O4SGGM"
140 B$="V2O4QEV3O4QCV1O4IGSGG"
150 C$="V1O5ICV2O4QEV3QCV1O5ID"
160 D$="V1O5IEV2O4QEV3QCV1O4SGG"
170 E$="V1O5ICV2O4QEV3QCV1O5IE"
180 F$="V1O5IDO4V2QFV3O3QGV1O4IB"
190 G$="V1O4IGV2O4QFV3O3QGV1O4SGG"
200 H$=LEFT$(D$,18)+"O5SCE"
210 I$="V1O5QGV2O4QFV3O3QG"
220 J$="V3QGV2O4QFV1O5SRSFED"
230 K$="V1ICV2O4QEV3QCV1O5IE"
240 L$="V1ICV2O4QEV3QC"
250 PLAYA$:PLAYB$:PLAYB$:PLAYC$:PL
    AYD$
260 PLAYB$:PLAYE$:PLAYF$:PLAYG$
270 PLAYB$:PLAYB$:PLAYC$:PLAYH$
280 PLAYI$:PLAYJ$:PLAYK$:PLAYL$
290 GOTO 250
```

SUPERMARKET

INNE (C-64) - dokończenie ze strony 29

■ Listowny Klub Użytkowników C-64. Informacje koperta + znaczek. Nowa Sól, 67-100, Oś. Konstytucji 8F/72.

INNE (Amiga)

■ Pilnie kupię programy na A500: Disk Coder 1.0, Kidwriter Applied Physics, Logic Works, Dos-2-Dos (oryginał) oraz wymienię gry, programy i doświadczenia. Kupię też gry: Ghostbusters I, Test Drive III, IV lub V, Spuer Bill - wyłącznie oryginał. Mariusz Gaik, 34-615 Słopnica 829, woj. nowosądeckie.

■ Sprzedam programowy emulator IBM (CGA) - 50000 zl, oraz książkę MC 68000 (J.Kostrzewski) - 100000 zl. Piotr Laszczyk, 34-511 Kościelisko, Szeligówka 976, tel. 70-444.

■ Sprzedam drukarkę LC-200 COLOR z wbudowanymi polskimi znakami. Cena 4 mln zł lub 290 $. Marcin Balewicz, ul. Odrzańska 10/103, 30-408 Kraków, tel. (012) 67-11-52.

■ Zamienię oryginalny, niemieckojęzyczny podręcznik "Workbench 2.0" z komputera A600 lub A500+ na jego wersję anglojęzyczną. T.Fleszar, Plac na Bramie 10A/3, 37-700 Przemyśl.

■ Kupię monitor kolorowy. Łukasz Nowaczyk, Oś. Jagiellońskie 36c/6, 62-200 Gniezno.

■ Sprzedam: Bajtki, układ scalony ADC 0809, pomogę w zrobieniu samplera do Amigi lub zrobię na zamówienie. Informacja: koperta + znaczek. Radosław Sapieja, ul. Gliniana 77/1, 50-526 Wrocław.

KLUBY I SZKOŁY

■ Redakcja "C&A" poszukuje informacji o szkołach o profilu komputerowym, klubach, kółkach zainteresowań i innych grupach skupiających użytkowników komputerów firmy Commodore. Informacje otrzymane tą drogą posłużą nam do sporządzenia listy, klubów i grup do których można zwrócić się z prośbą o radę i pomoc. W wypadku szkół chcielibyśmy przygotować listę placówek (podstawowych i ponadpodstawowych) w których można kontynuować swoje zainteresowania. Listy (z dopiskiem KLUBY) prosimy kierować do redakcji podając dokładny adres placówki, typ(y) wykorzystywanego sprzętu, profil działalności oraz imię i nazwisko osoby z którą należy nawiązać kontakt.



AUTORUN INACZEJ

Program AUTORUN opublikowany w C&A nr 3/92 jest bardzo ciekawy, tyle że jeżeli ktoś nie ma dostępu do gotowej, dyskowej wersji programu, może mieć kłopot z wpisaniem go do komputera. W tym artykule chciałbym więc opisać najprostszy chyba sposób na samouruchamianie programów. Uwagi te jednak proszę potraktować jedynie jako ciekawostkę czy wprowadzenie do „prawdziwego" samouruchamiania, bo prawdę mówiąc, nie jest to dobry sposób na zabezpieczenie programu, a wspominany wcześniej AUTORUN z C&A 03/92 jest o niebo lepszy.

Najpierw musimy napisać tzw. loader, czyli procedurę, której zadaniem jest wczytanie programu głównego. Najprościej będzie skorzystać z procedury systemowej LOAD. Dostępna jest ona bezpośrednio pod adresem $F49E, poprzez tablicę skoków Kernal $FFD5 oraz poprzez wektor systemowy zawarty w komórkach $0330 i $0331. Przed wywołaniem należy do akumulatora wpisać wartość 0 (jedynka spowodowałaby wykonanie weryfikacji), a do rejestrów X i Y młodszy i starszy bajt adresu, pod który dane mają być wczytywane. Przy założeniu, że jest to adres $1000 nasza procedurka wyglądałaby następująco:

```
LDA #$00
LDX #$00
LDY #$10
JSR $FFD5
```

Zanim jednak wpiszesz i uruchomisz ten program (za pomocą monitora języka maszynowego lub makroasemblera), trzeba jeszcze ustalić parę rzeczy. Najpierw musimy określić nazwę pliku przeznaczonego do wczytania. W tym celu do akumulatora wpisujesz liczbę znaków nazwy, a do rejestrów X i Y adres, pod którym ta nazwa się znajduje. Następnie należy wywołać procedurę SETNAM za pośrednictwem tabeli skoków (JSR $FFBD).

Pozostaje nam jeszcze do wykonania procedura SETLFS (otwarcie zbioru logicznego). Musisz w tym celu wpisać do akumulatora numer pliku. Masz tu dosyć dużą dowolność, ale najlepiej wpisać wartość 1. W rejestrze X umieść numer urządzenia (01 dla magnetofonu, 08 dla stacji dysków). Rejestr Y musi zawierać adres wtórny lub liczbę $FF. Adres wtórny może przybierać wartości 0 lub 1. Zastosuj 1, jeśli chcesz wczytać program pod określony adres w pamięci lub 0, jeśli jest to dla Ciebie obojętne. Procedura SETLFS dostępna jest za pośrednictwem tablicy skoków pod adresem JSR $FFBA. Całość wygląda tak, jak to przedstawiono na listingu 1.

Cały loader najlepiej jest wpisać od adresu $02A7 (musi zmieścić się do $02FF), korzystając z dowolnego monitora języka maszynowego. Teraz należy zmienić wektor pętli głównej interpretera BASIC tak, aby wskazywał na początek naszego loadera. Operację tę przeprowadzamy wpisując do komórek pamięci $0302 i $0303 odpowiednio wartości $A7 i $02. Całość zapisz na dyskietce (lub taśmie) poleceniem:

S „nazwa", 08, 02A7, 0304

lub instrukcją właściwą dla używanego przez Ciebie monitora. W ten sposób na dyskietce (lub kasecie) zostanie zapisany gotowy plik „nazwa", który wczytujemy rozkazem LOAD „nazwa", 8,1. Uruchomi się on automatycznie, bowiem system po zakończeniu wczytywania, chcąc wrócić do BASIC, natrafi na zmieniony wektor w komórkach $0302 i $0303 i uruchomi Twój loader.

Podkreślam jeszcze raz, że jest to najprostsza forma samouruchamiania się. Nie jest na pewno zbyt efektowna; o wiele lepsze efekty daje wykorzystanie stosu mikroprocesora. Metodę tę przedstawię za miesiąc.

MARCIN „FOX" LIS

```
LDA  #$        ; długość
               ; nazwy pliku
LDX  #$        ; młodszy bajt
               ; adresu nazwy
LDY  #$        ; starszy bajt
               ; adresu nazwy
JSR  $FFBD
LDA  #$01
LDX  #$        ; numer
               ; urządzenia
LDY  #$        ; adres
               ; dodatkowy
JSR  #$FFBA
LDA  #$00
LDX  #$        ; młodszy bajt
               ; adresu
               ; wczytywania
LDY  #$        ; starszy bajt
               ; adresu
               ; wczytywania
JSR  $FFD5
JMP  $         ; początek
               ; programu
               ; głównego
```

POZNAJ SIŁĘ FONTMASTERA

czyli edytorów tekstu ciąg dalszy

W poprzednim numerze C&A mój redakcyjny kolega, Bartek Dramczyk, opisał jeden z popularnych edytorów tekstu na C-64 — POLSCRIPT. Edytor, do którego używania postaram się Was zachęcić niniejszym artykułem, jest z pewnością mniej znany, ale za to można go bez przesady określić mianem „profesjonalny", zwłaszcza, jeśli chodzi o jakość druku.

FONTMASTER II ma naprawdę imponujące możliwości: posiada aż 71 poleceń kontrolujących pracę drukarki (bądź wpływających pośrednio na ostateczny wygląd druku), znakomicie współpracujące ze stacją dysków (dalszych kilkanaście poleceń), umożliwia drukowanie tekstów w kilku językach jednocześnie, potrafi drukować tekst szpaltowany, pozwala tworzyć własne zestawy znaków i czcionek, bezproblemowo dokonuje też konwersji plików z innych edytorów tekstu na zrozumiały dla siebie format. Ale zacznijmy od początku.

Po wczytaniu loadera, na ekranie ukazuje się menu systemowe. Mamy do wyboru: Word processor (edytor tekstu), Font creator (program do tworzenia własnych zestawów znaków dla drukarki), Character set creator (do projektowania znaków ekranowych), File translator (translator tekstów) i Setup (pozwala odpowiednio skonfigurować całość).

Gdy korzystamy z programu po raz pierwszy (i tylko wtedy), należy naturalnie wybrać Setup, aby poinformować FONTMASTERA, jakim sprzętem dysponujemy. Chodzi tu głównie o drukarkę i interfejs, chociaż można też zmienić numer urządzenia (8–11) dla stacji dysków i ustalić wg gustu kolory ramki, tła i tekstu. Nie martwcie się, jeżeli nie znajdziecie w spisie swojej drukarki. Większość tych urządzeń używa tych samych lub podobnych kodów sterujących, tak więc wystarczy, że wybierzecie drukarkę o parametrach zbliżonych do Waszej. Wyboru można zresztą dokonywać na ślepo korzystając z opcji „Test this Setup"; umożliwia ona natychmiastowe sprawdzenie poprawności pracy drukarki. Po ustawieniu wszystkich parametrów należy wskazać kursorem napis „Save this Setup" i nacisnąć RETURN. Podczas wczytywania edytora, dane wczytywane są do pamięci automatycznie.

Przejdźmy teraz do edytora (Word processor). Wczytanie go do pamięci zajmuje komputerowi ok. 90 sekund. Cztery górne wiersze planszy roboczej zawierają ekran pomocniczy (Help screen) wyświetlający na bieżąco bardzo wiele (w porównaniu np. z POLSCRIPTEM) informacji. Szczególnie użyteczne są wskaźniki czcionek (fontów) aktualnie rezydujących w pamięci oraz ściągawka z klawiszologii. „Help Screen" można wyłączyć za pomocą klawiszy CTRL i H.

Tekst wpisujemy jednym ciągiem, tzn. używamy klawisza RETURN tylko na końcu akapitu, a nie, jak w przypadku POLSCRIPTA, na zakończenie każdej linii. Do poruszania się w tekście służą, oprócz kursora, następujące kombinacje klawiszy:

CTRL B — przejście na koniec tekstu,
CTRL kursor w dół — następna strona,
CTRL kursor w górę — poprzednia strona,
HOME (raz) — początek „strony" (ekranu)
HOME (dwa razy) — początek tekstu,
F1 — następne słowo,
F3 — następne zdanie,
F5 — następne polecenie formatujące,
F7 — następny modyfikator.

Klawisze funkcyjne działają w „tył", po ich wciśnięciu z klawiszem SHIFT. Spośród ułatwień edycyjnych wyróżnić można:

CTRL E — usunięcie tekstu od kursora do końca,
CTRL + — ustawienie tabulatora,
CTRL - — usunięcie tabulatora,
CTRL SHIFT - — usunięcie wszystkich tabulatorów,
CTRL DEL — usunięcie linii tekstu,
CTRL SHIFT INST — wstawienie pustej linii,
CTRL SHIFT R — reguluje szybkość przesuwu kursora i reakcji komputera na naciskanie klawiszy (czasem bardzo przydatne),
CTRL I — włączenie/wyłączenie trybu INSERT (wstawianie),
CTRL F — wyszukiwanie sekwencji znaków,
CTRL R — wyszukiwanie sekwencji znaków z zamianą na inną.

Jako ułatwienie przy określaniu sekwencji do odszukania (i ewentualnej zamiany) służy pytajnik (zastępuje znak) oraz nawias kwadratowy, pełniący rolę ogranicznika. Jeśli np. sekwencję do znalezienia zdefiniujemy jako „ma", szukanie zakończy się na słowie np. „mama", ale nie na „rama".

Równoczesne naciśnięcie klawiszy CTRL i M służy do zaznaczania początku bloku tekstu. Po „dojechaniu" kursorem na koniec bloku mamy do dyspozycji następujące polecenia:

CTRL E — usunięcie bloku,
CTRL K — przeniesienie bloku do bufora bez usuwania,
CTRL C — przeniesienie bloku do bufora z jednoczesnym usunięciem go z tekstu,
CTRL P — skopiowanie bloku z bufora.

Blok nie może zawierać więcej niż 1280 znaków. W przypadku przekroczenia tego limitu ukazuje się komunikat „Block is too large". Należy też pamiętać, że użycie opcji „Video preview" (o której niżej) oraz drukowanie wymazuje zawartość bufora.

Polecenia kontrolujące pracę drukarki dzielą się na dwa typy: modyfikatory oraz polecenia formatujące tekst. Modyfikatory wpisuje się bezpośrednio w tekście, naciskając odpowiedni klawisz wraz z klawiszem Commodore Logo (znak Commodore, lewy dolny róg klawiatury). Na ekranie pojawi się wówczas charakterystyczny znaczek w rewersie.

Działanie modyfikatora kończy najczęściej ponowne jego użycie, czasem jednak trzeba użyć modyfikatora o przeciwnym działaniu. A oto lista modyfikatorów FONTMASTERA (znak po lewej oznacza, jaki klawisz należy nacisnąć równocześnie z klawiszem C=):

a — wybór gęstości druku 11 CPI (11 zn./cal, *alt pitch*),
b — włącza/wyłącza tłusty druk (*boldface*),

c — włącza/wyłącza dwukrotnie ścieśnienie druku (*compress*),
e — wybór gęstości druku 12 CPI (*elite pitch*),
i — włącza/wyłącza rewers (*inverse*),
k — wybór gęstości druku 13 CPI (*konnect pitch*),
m — dwukrotnie mniejsza wysokość druku (*micro height*),
n — normalna wysokość druku (*normal height*),
o — nakładanie na siebie dwóch znaków w tym samym miejscu (*overlay*; pozwala tworzyć znaki specjalne),
p — wybór gęstości druku 10 CPI (*pica pitch*),
t — dwukrotnie większa wysokość druku (*tall height*),
u — włącza/wyłącza podkreślanie (*underlining*),
x — włącza/wyłącza druk rozszerzony (*expand*),
↑ — następny znak w indeksie górnym (*superscript*),
↓ — następny znak w indeksie dolnym (*subscript*),
↑ i SHIFT — włącza indeks górny na stałe,
↓ i SHIFT — włącza indeks dolny na stałe,
RETURN — wyłącza oba indeksy,
1 (do **9**) — wybór kroju czcionki ze slotu o numerze 1 (do 9).

Modyfikatory można ze sobą „mieszać", co daje oczywiście ogromne możliwości. Weźmy np. gęstość druku: użycie modyfikatorów **e** i **x** daje w efekcie gęstość druku 6 CPI, a np. **k** i **c** — 26 CPI.

Do wpisywania poleceń formatujących służy klawisz ze znakiem funta (£). Po jego naciśnięciu na ekranie pojawia się pogrubiona strzałka, za którą należy podać polecenia (rozdzielone dwukropkiem), a na końcu nacisnąć RETURN. Ze względu na miejsce omówię tu tylko najważniejsze znaki formatujące tekst:

PLxx — (*paper length*); długość strony (kartki papieru). Chociaż w instrukcji obsługi podano, że parametr xx równy jest długości strony w calach pomnożonej przez 6 (co daje 69 dla kartki formatu A4), to z mojego doświadczenia wynika, iż należy tu przyjąć za mnożnik cyfrę 7 (a więc stronę A4 definiuje PL80).
TMxx — (*top margin*); górny margines (pomiędzy brzegiem kartki i pierwszą linią tekstu).
BMxx — (*bottom margin*); dolny margines (pomiędzy brzegiem kartki i ostatnią linią tekstu).
LMxx — (*left margin*); lewy margines. Mamy do wyboru kilka wariantów tego polecenia:
LMxx — lewy margines wg parametru xx;
LM+x — lewy margines przesunięty w prawo o x znaków;
LM-x — jak wyżej, ale w lewo;
LM+xT — jednorazowe przesunięcie w prawo o x znaków; po napotkaniu pierwszego znaku odpowiadającego klawiszowi RETURN, FONTMASTER ustawi lewy margines w poprzednią pozycję;
LM-xT — jak wyżej, ale w lewo.
RMxx — (*right margin*); określa pozycję prawego marginesu na zasadach opisanych powyżej.
INxx — (*indentation*); wcięcie tekstu. Pozwala tworzyć akapity przesunięte zarówno w prawo (np. **IN5**), jak i w lewo (np. **IN-7**). FONTMASTER tworzy akapit automatycznie po każdym użyciu RETURN.
JS — (*justification*); równanie tekstu do obu marginesów).
WW — (*word wrap*); wyrównywanie tekstu do lewego marginesu.
ER — (*edge right*); wyrównywanie tekstu do prawego marginesu.
CY — (*centering on*); pośrodkowanie tekstu (**CN** — wyłączenie tego trybu).
PY — (*proportional on*); druk proporcjonalny (**PN** — wyłączenie).
CSxx — (*character spacing*); umożliwia dowolne określenie odstępu między literami. Jednostką dla parametru xx jest tu średnica igły drukarki. Polecenia tego nie musimy oczywiście używać, gdyż odstęp między literami sterowany jest normalnie modyfikatorami (a, e, p, k). Dla druku standardowego (pica) odstęp ten wynosi 3, dla druku o gęstości 13 CPI (konnect) — 0.
LSxx — (*line spacing*); odstęp między wierszami tekstu (wyrażany w wierszach na cal).
C#x — określa liczbę szpalt. Można podzielić stronę na maksimum cztery szpalty. Tekst pierwszej szpalty drukowany jest automatycznie do końca strony, po czym drukarka cofa papier i następuje wydruk kolejnej szpalty. Do określenia marginesów szpalt 2–4 (marginesy pierwszej szpalty określane są poleceniami LM i RM) służą analogiczne polecenia: 2L i 2R, 3L i 3R oraz 4L i 4R. Należy pamiętać, aby marginesy poszczególnych szpalt nie zazębiały się. W przypadku, gdy drukarka „nie potrafi" cofnąć papieru, wydruk zostaje zatrzymany a na ekranie pojawia się komunikat nakazujący ręczne cofnięcie.
SS — (*single sheet*); przerywa wydruk w momencie osiągnięcia końca strony. Polecenie to służy do wydruku tekstu na pojedynczych kartkach.
NP — (*new page*); następny wiersz po tym poleceniu zostanie wydrukowany na nowej stronie.
SOxx — (*sub/superscript offset*); umożliwia dowolne określenie pozycji pionowej dla dolnych/górnych indeksów. Wartość standardowa — 6.

FONTMASTER jest również wyposażony w cały zestaw wpisywanych analogicznie poleceń przeznaczonych specjalnie do formatowania stopek (footer) i nagłówków (header). Są to m.in.:

HPxx — (*header position*); pozycja (odstęp) nagłówka względem tekstu. Wartość początkowa — HP4.
FPxx — jak wyżej, ale dotyczy stopki.
HLxx — lewy margines nagłówka i stopki.
HRxx — prawy margines nagłówka i stopki.
H= — służy do definiowania nagłówka.
F= — służy do definiowania stopki.
P#xx — ustalenie numeru pierwszej drukowanej strony (następne numerowane są automatycznie).
RO — rzymska numeracja stron (tryb ten wyłącza się poleceniem **DC**).

Podczas definiowania stopki/nagłówka mamy do dyspozycji prawie wszystkie modyfikatory, wpisujemy je jednak nie jak w „normalnym" tekście, lecz używamy komputerowego znaku dzielenia. Wygląda to np. tak:

H=/b — (nagłówek będzie drukowany tłustym drukiem),
albo
F=/r — (wyrównanie stopki do prawego marginesu) itp.

Jak widać, litery oznaczające dany modyfikator są identyczne z poprzednimi, a nieco inna postać modyfikatorów pozwala na oddzielne, nie wpływające na właściwy tekst, formatowanie stopki czy nagłówka.

Jeśli nie użyjesz żadnego polecenia dotyczącego stopki bądź nagłówka, zostaną one wydrukowane pismem standardowym (pica), czcionką przypisaną do pierwszego slotu (o slotach za chwilę) i wypośrodkowane względem tekstu. Automatyczne numerowanie stron uzyskać można dzięki poleceniu H(F)□ (w miejscu kwadratu zostanie wydrukowany numer strony; kwadrat ukazuje się po naciśnięciu CTRL i V).

Najważniejszą cechą FONTMASTERA, od której zresztą edytor ten wziął swą nazwę (ang. font — czcionka), jest możliwość drukowania tekstów dowolną czcionką a także wydruk dowolnych znaków (np. matematycznych). Na dyskietce z FONTMASTEREM znajdują się 33 kroje (m.in. cyrylica, hebrajski, grecki). Dzielą się one na „normalne" i „superfonty", których wygląd nie odbiega jakością od profesjonalnego druku. Aby skorzystać z jakiejkolwiek czcionki trzeba ją najpierw wczytać do wydzielonych specjalnie dla nich obszarów pamięci, tzw. slotów, których jest dziewięć. Jeden dokument może więc być drukowany aż dziewięcioma krojami czcionki (normalnych, bo superfont zajmuje dwa sloty). Aby dany odcinek tekstu został wydrukowany odpowiednią czcionką, wystarczy użyć modyfikatora (klawisz C= i cyfra w zakresie 1—9).

Dla wybrednych użytkowników przewidziano możliwość tworzenia własnych czcionek. Służy do tego program Font creator wczytywany z menu głównego. Pozwala on zarówno na tworzenie całkiem nowych

czcionek i znaków, jak i na modyfikację już istniejących. Samodzielne dorobienie polskich znaków diakrytycznych do dowolnej czcionki nie przedstawia najmniejszego problemu.

Oczywiście tak duże możliwości FONTMASTERA wynikają z tego, iż edytor ten drukuje w trybie graficznym — drukowanie odbywa się więc raczej wolno (a dla niecierpliwych: bardzo wolno). Istnieje co prawda możliwość druku w trybie znakowym (tryb ten można włączyć/ /wyłączyć poprzez naciśnięcie SHIFT, CTRL i D), lecz wówczas nie działa większość modyfikatorów i poleceń formatujących, jakość pozostawia wiele do życzenia (draft), a ponadto w niektórych drukarkach konieczne jest jeszcze odpowiednie skonfigurowanie drukarki. Nie ma również mowy o polskich bądź jakichkolwiek innych znakach nie występujących w alfabecie łacińskim, chyba, że drukarka pozwala na wczytanie polskich znaków i ich „zablokowanie" w pamięci drukarki.

Normalnie FONTMASTER pozwala tworzyć dokumenty zajmujące do 539 linii. Istnieje jednak możliwość poszerzenia obszaru pamięci przeznaczonego na tekst. W tym celu należy ograniczyć pamięć wykorzystywaną na przechowywanie czcionek (kombinacja CTRL, SHIFT i S, po czym należy wcisnąć cyfrę określającą z ilu krojów pisma będziesz korzystać). W ten sposób można uzyskać dokumenty liczące sobie aż 900 linii. Użytkownik, naciskając klawisze CTRL i „?", może w każdej chwili sprawdzić, ile jeszcze zostało wolnej pamięci, ponadto istnieje procentowy wskaźnik zajętości pamięci wyświetlany w linii statusowej u góry ekranu.

O klasie FONTMASTERA decyduje też opcja umożliwiająca użytkownikowi wstępne przejrzenie tekstu (*video preview*) przed wydrukowaniem w postaci zbliżonej do tej, jaką ujrzymy na papierze, a konkretnie:

— tekst wyświetlany jest w trybie graficznym 80 znaków w wierszu;
— zachowany jest rzeczywisty podział na strony i numeracja stron;
— nie są wyświetlane znaki specjalne (polecenia formatujące, modyfikatory);
— tekst wyróżniony (np. dwa razy wyższy, albo pisany tłustym drukiem) wyświetlany jest, dla przypomnienia, w rewersie.

Dzięki opcji video preview oszczędza się sporo czasu i/lub papieru.

FONTMASTER współpracuje tylko ze stacją dysków. Kilkanaście poleceń, zaspokajających absolutnie wszystkie potrzeby, dostępnych jest po naciśnięciu CTRL i D. Tu wartu zaznaczyć, że FONTMASTER tworzy dwa rodzaje plików: PRG i SEQ. Pliki PRG zawierają oprócz właściwego tekstu informacje odnoszące się do użytych w dokumencie czcionek i tabulatorów, co dodatkowo ułatwia pracę (żądane kroje są wczytywane automatycznie w odpowiedniej kolejności poprzez naciśnięcie ledwo dwóch klawiszy — CTRL i N). Pliki SEQ pozbawione są tej zalety, dają się za to łatwo przenosić na inne komputery jako zbiory ASCII.

I ostatnia istotna sprawa: FONTMASTER „przyjmuje" pliki pochodzące z innych edytorów tekstu, takich jak SPEEDSCRIPT, EASY SCRIPT, SCRIPT64, PAPERCLIP, FLEET SYSTEM 2, OMNIWRITER, CREATIVE WRITER i FONTMASTER I.

Ogólnie rzecz biorąc należy przyznać, że edytor ten jest znakomity, jednakże stawia wysokie wymagania przede wszystkim drukarce; najlepsze wyniki można osiągnąć mając drukarkę pozwalającą na wydruk grafiki z poczwórną gęstością, co spełniają np. wszystkie drukarki Star od modelu NL-10 w górę.

ARNOLD

FONTMASTER II (Commodore 64, dyskietka)
Autor: Marty Flickinger
XETEC, 1985.

Przykładowe możliwości FONTMASTERA:

Druk rozszerzony.

Druk zgęszczony, druk dwa razy wyższy. druk "mikro".

Druk tłusty, tłusty i rozszerzony.

Druk "mikro" i rozszerzony.

Indeks górny i dolny - "elite pitch".

"Alt pitch", "connect pitch".

Druk dwa razy wyższy, rozszerzony, podkreślony i tłusty.

A teraz niektóre fonty:

shadow, bauhaus, tech, כצגשהד ירה (hebrajski), split, русский, script, *kursywa*.

SUPERMARKET

COMMODORE 64

■ Sprzedam C-64II (gwarancja), magnetofon, joystick, FINAL III, 40 kaset z grami i programami, literaturę za 2300000 zł. Tomasz Szatewicz, Oś. Mazurskie 18/17, 11-700 Mrągowo.

■ Pilnie sprzedam: C-64C, stację 1570, drukarkę D100M, magnetofon, moduły, 2 joysticki, oprogramowanie użytkowe i gry (dyskietki, kasety), obszerną literaturę. Cena około 4.8 mln zł. Tomasz Pieszyński, Łódź, ul. Killńskiego 118/16, tel. 74-48-88.

■ Sprzedam C-64, magnetofon, X, 22 kasety z grami (1.5 mln zł). Kamil Ratajczak, ul. Buczka 4/13, 14-100 Ostróda.

■ Sprzedam "Bazę Danych" dla C-64 (dysk, instrukcja, dodatkowy program). Cena: 25000 zł. Dominik Latusek, Oś. Kosmonautów 23/4, 61-642 Poznań.

■ Sprzedam C-64II, magnetofon, gwarancja, joystick, BLACK BOX III, pokrywa, 500 programów, literaturę. Cena: 2 - 2.5 mln zł. A. Tomulewicz, ul. Stroma 35/8, 15-662 Białystok, tel. 613-362.

■ Sprzedam C-64II (gwarancja), magnetofon, Black Box. Do zestawu dołączę ok. 400 gier i programów. Cena: 2 mln zł. Piotr Kowalczyk, 02-601 Warszawa, ul. Racławicka 42/92.

■ Sprzedam C-64II z osprzętem, literaturą i oprogramowaniem. D. Cierpiał, ul. Bankowa 7/18, 42-290 Blachownia k/Częstochowy, tel. 328-28.

■ Pilnie sprzedam C-64 II (stan idealny), magnetofon, 1571, Final II, X-Plus, 3 joysticki, 400 gier (w tym 60 dyskietek), literaturę. Cena: 5 mln zł. Tomasz Weber, ul. Grunwaldzka 35A, 34-330 Żywiec.

■ Sprzedam C-64 II, 1541 II, 1535, Final III, Black Box III, 2 joysticki, gry na dyskietkach i kasetach i literaturę (4 mln zł). Krzysztof Kliman, ul. Wojska Polskiego 12/14, 66-620 Gubin.

■ Sprzedam C-64 II (gwarancja), magnetofon, monitor 1802 (gwarancja), joystick, Final II, Black Box, X, 280 gier i literaturę za 5 mln zł. T. Basiński, ul. Dunikowskiego 15A/1, 80-528 Gdańsk, tel. 43-53-77.

■ Sprzedam C-64 II (stan idealny), 1530, 2 joysticki, 2 moduły, około 750 programów, pokrywę, literaturę. Cena ok. 1.8 mln zł. Hubert Gajewski, ul. J.Pawła II 27/7, 09-200 Sierpc, tel. 75-38-23.

■ Zamienię mini-wieżę Vonix (gwarancja) na używaną stację 1541 II (z dopłatą). Adam Ostach, ul. Słowackiego 18/26, 62-300 Wrześnie, tel. 361-128.

■ Kupię drukarkę do C-64 (np. Star LC-10). Marcin Ślusarczyk. ul. Czachowskiego 11/19, 27-600 Sandomierz.

■ Kupię pióro świetlne do C-64. Łukasz Grochulski, ul. Maratońska 23/16, 94-102 Łódź, tel. 87-84-11.

■ Sprzedam stację dysków 1541 II wraz z dyskietkami (gry i użytki), literaturę za 2.5 mln zł. Michał Rzodkiewicz, ul. Księżodworska 68, 13-200 Działdowo.

■ Zamienię zielony, firmowy, monitor Commodore na dowolną stację do C-64. Piotr Bonifaciuk, ul. Miłosna 1/19, 08-300 Sokołów Podlaski.

■ Poszukuję stacji dysków do C-64 na raty oraz pism, broszur, książek itp. na temat C-64. Zbigniew Leśniewski, Oś. Południe 6/25, 19-203 Grajewo.

■ Zamienię magnetofon 1530, 4 moduły, 200 gier i programów użytkowych + dopłata ok. 400 tys. zł na dobrą i używaną stację 1541 do C-64. R. Karczmarczyk, ul. Szopena 48c/2, 41-400 Mysłowice.

■ Sprzedam C-64, magnetofon, 2 joysticki, 400 programów, Black Box III, pokrywę, literaturę. Cena 2.8 mln zł. Piotr Oskwarek, ul. Kombatantów 1, 56-300 Milicz.

AMIGA

■ Zamienię/sprzedam ATARI 65 XE (magnetofon, joystick, kasety, literatura) na rozszerzenie RAM do Amigi (512 kB), modulator lub stację 5,25". Marek Mazłowski, ul. Wielka 13/4, 41-200 Sosnowiec.

■ Zamienię na używaną Amigę: roczny C-64 (stan idealny), magnetofon, ok. 800 programów i gier, moduł Black Box V.2. Michal Jaśkiewicz, Gronowo Górne 9/6, 82-300 Elbląg.

■ Sprzedam lub zamienię na Amigę 2000 (z dopłatą): C-64C (nowa wersja), 1541 II (gwarancja), magnetofon, Final III, moduł z grami, 40 dyskietek firmowych i 10 kaset z około 600 programami, literaturę. Cena 5.1 mln zł. Możliwość zdekompletowania zestawu. Jacek Stolarski, ul. Zbiegniewskiej 13/93, 87-800 Włocławek, tel. 321221 w.192.

■ Sprzedam Amigę 500, chip memo, 1 MB, monitor kolorowy 1084S, filtr, drukarkę LC-20, 150 dysków, TOP STAR, literaturę, cena 17 mln zł. Całość to sprzęt nowy do 2 miesięcy. Wojciech Wasielewski, ul. Grunwaldzka 12b/52, 99-301 Kutno, tel. 365-78.

■ Sprzedam Amigę 500 (1 MB), modulator, dyskietki, literaturę, monitor 1084S. K. Kwiatek, ul. Do Studzienki 61, 80-227 Gdańsk, tel. 419-613.

■ Sprzedam Amigę 500 (1 MB), monitor kolorowy 1084S (gwarancja), modulator, 140 nagranych dysków z pudełkiem, joystick, pokrywę, literaturę. Radosław Wesołowski, ul. Doleńców 26, Szczecin, tel. 620-326.

■ Sprzedam Amigę 500 (gwarancja), monitor kolorowy Commodore 1084S (gwarancja), joystick, dyskietki. Stan idealny. Hubert Gąsior, 09-407 Płock, ul. Lachmana 2/22, tel. 366-01.

■ Zamienię C-64, stację 1541 II (gwarancja 2 lata), 1530, Black Box, dyskietki, kasety na Amigę 500. Tomasz Dzioba, Oś.XXX-lecia 4/43, 11-500 Giżycko.

■ Sprzedam modulator TV do Amigi 500 - gwarancja - tanio. Adam Deptolla, ul.Opalenicka 2, 64-310 Lwówek Wlkp., tel. 260.

INNE (C-64)

■ Mam C-64 i magnetofon. Wymienię gry. Poszukuję: Wings of Fury 3, Darks Challenge, Rally Africa, Tokio Race. Wymienię też dema. Zamienię lub sprzedam (90000 zł) moduł Black Box III na Ex-Plus, X, Expert, Warsaw BASIC lub inne. Krzysztof Gołąb, ul. Centaura 31/15, 44-117 Gliwice.

■ Kupię gry: Navy S.E.A.L., War in the Crown, Spike in Transylvania, ACE, Solo Flight, F-19 Stealth Fighter (na kasetach) orsz interpreter Simon's BASIC dla C-64. Krzysztof Wolos, ul. Jesionowa 3, 37-700 Przemyśl.

■ Wymienię oprogramowanie C-64 (2500 pozycji) i C-128 (30 pozycji). K.Gadacz, ul. Świętojańska 16/5 62-500 Konin.

■ Poszukuję gier: Defender of the Crown, G.I.Joe, Match Fishing i wszelkich karate (C-64, magnetofon) w zamian za innych 160 gier. Sebastian Kołodziej, ul. Gromady Grudziąż 21/68, 30-657 Kraków.

■ Wymienię oprogramowanie dla C-64, C-128, wyłącznie na dyskietkach. Mam C-128 i stację 1541 II. Krzysztof Karbowniczyn, Al. Lotników Polskich 34/8, 21-040 Świdnik.

■ Poszukuję programu do nauki języka niemieckiego (Commodore 64, kaseta). Zwrócę nośnik. Artur Drążyk, ul. Pułaskiego 10m10, 03-400 Otwock.

■ Sprzedam tanio gry na C-64 w wersji kasetowej. Cena wszystkich 220 gier: 85000, cena jednej: 2500 zł, cena kasety (20 gier): 28000. Podam spis. W.Walczak, ul. Wrocławska 73/13, 63-200 Jarocin.

■ Wymienię oprogramowanie dla Commodore 64. Mam ok. 500 tytułów. Mariusz Sitarek, ul. Fonologicana 3B/53, 96-100 Skierniewice.

■ Kupię literaturę dla C-64: C-64 Programmer's Reference Guide, Mapping The Commodore 64, The C-64 Puzzle Book, Adventure Games For The C-64, Jak rozbudować interpreter. Adam Wolski, ul. Wyzwolenia 51c/21, 80-537 Gdańsk.

■ Literatura do C-64, m.in. mapa pamięci (zawsze aktualna). Wiadomość: koperta + znaczek, skrytka pocztowa. 39, 11-600 Węgorzewo.

SPOSÓB NA BLACK BOX

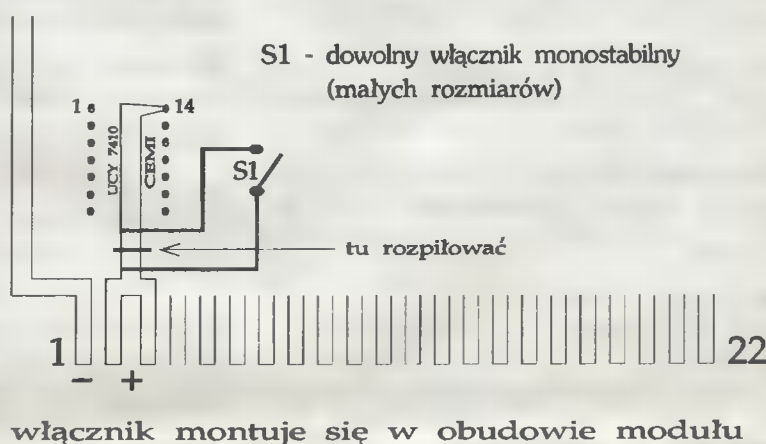
Jestem stałym Czytelnikiem C&A, posiadam C-64 II oraz moduł BLACK BOX wersja 4.0.

Po kupnie C-64 i modułu spotkało mnie rozczarowanie, a mianowicie miałem problemy z grą Montezuma's Revenge, którą uwielbiam. Gra po uruchomieniu działała poprawnie, aż do momentu wciśnięcia spacji (po planszy tytułowej): program zamiast uruchomić się, zaczynał „wariować", obraz znikał, po czym zgłaszał się moduł, jak gdyby nic się nie stało. Podobne problemy zauważyłem u moich kolegów posiadających BLACK BOX tak w wersji 4.0, jak i 8.0. Producent uczciwie zaznacza jednak, że nie wszystkie programy będą działać z modułem prawidłowo.

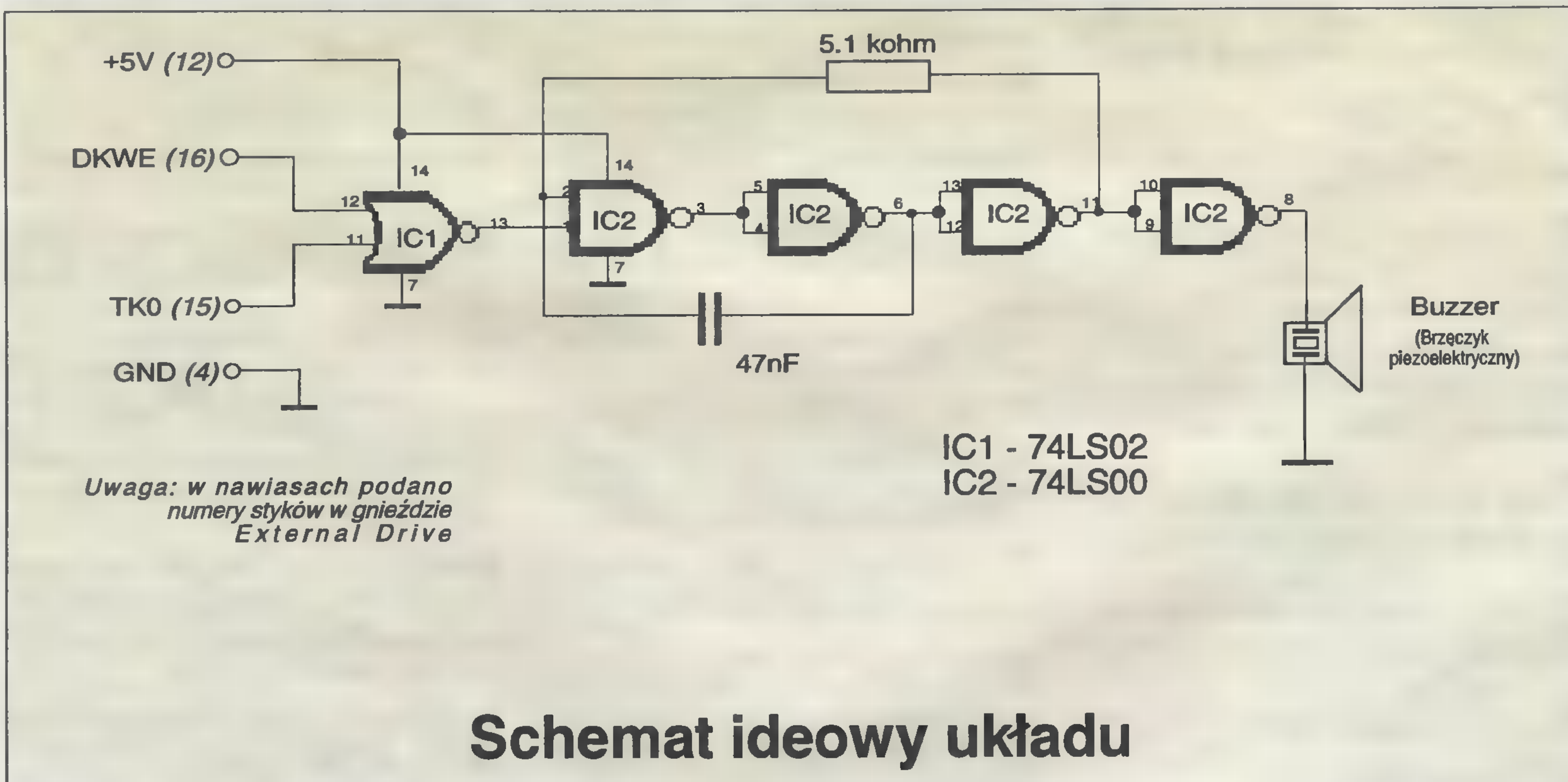
Ale do rzeczy. W swoim module zainstalowałem wyłącznik, który odcina zasilanie (+). Jest to rozwiązanie bardzo proste i skuteczne. Kiedyś, aby uruchomić Montezuma's Revenge, musiałem wyjmować moduł, wczytać TURBO itd. Jak wiadomo, wymaga to trochę manipulacji. A teraz zwyczajnie wczytuję tę grę i po uruchomieniu, podczas gdy na ekranie widnieje plansza tytułowa, wyłączam zasilanie modułu, dzięki czemu program działa bez zarzutu.

Ta drobna modyfikacja może czasem ułatwić uruchamianie niektórych programów, nie daje też żadnych skutków ubocznych dla komputera.

KRZYSZTOF STEFANOWICZ (lat 14)



Schemat „Hardvare Virus Protector"



```
;LISTING 1

lea      NazwaBiblioteki,A1   ;adres
                       ; nazwy biblioteki
move.l  $4,a6          ; adres tabeli
                       ; skoków biblioteki
                       ; "EXEC"
jsr     -408(a6)       ; wywołanie
                       ; procedury
tst.l   d0             ; czy wartość w
                       ; D0=0 ?
beq.w   Błąd           ; jeśli tak, to
                       ; koniec programu
move.l  d0,DosBase     ; jeśli nie, to
                       ; zapamiętaj
                       ; podaną wartość
....                   ; tu dalsza część
                       ; programu
Błąd:   rts            ; wystąpił błąd,
                       ; wyjdź z programu
NazwaBiblioteki: dc.b  'dos.library',0
DosBase:           dc.l    0
```

```
; LISTING 2

START:

lea      DosName,a1
move.l  $4,a6
jsr     -408(a6) ; wywołanie
            ;"OldOpenLibrary"
tst.l   d0
beq.w   Blad
move.l  d0,DosBase
move.l  d0,a6
move.l  #Okno,d1     ; adres
       ; parametrów dla okna
move.l  #1005,d2
jsr-30(a6) ; wywołanie "Open"
tst.l   d0
beq.w   Blad
move.l  d0,OknoHD
move.l  DosBase,a6
move.l  d0,d1
move.l  #Tekst,d2 ; początek
                  ; tekstu
move.l  #TekstEnd-Tekst,d3
            ; długość naszego
            ; tekstu
jsr     -48(a6)   ; wywołanie
                  ; "Write"

Blad:   rts
DosName: dc.b 'dos.library',0
DosBase: dc.l    0
OknoHD:  dc.l    0
Tekst: dc.b 'Tu jest tekst!',0
TekstEnd: even
Okno:  dc.b 'CON:100/50/320/10
                0/* Okno *',0
```

Liczba kolejnych zeszytów / Tytuł	3	6	12	po ile egz.
Bajtek	X	60000	120000	
C&A	30000	60000	X	
TOP SECRET	27000	54000	X	

Wpłat dokonywać na konto:
Spółdzielnia "BAJTEK"
Bank "Agrobank S.A."
470005-1834-131
ul. Grochowska 262
04-398 Warszawa

tu zanotuj, co zamówiłeś

Warunki prenumeraty:

- Prenumerata zawarta przed upływem ważności kuponu gwarantuje niezmienność cen
- Przesyłka pocztowa nie wymaga dodatkowych opłat
- Jeżeli w ciągu 2 tyg. od pojawienia się numeru w kioskach przesyłka nie nadeszła, prosimy o kontakt
- Za błędy wynikające z niestarannnego wypełnienia formularza redakcja nie ponosi odpowiedzialności
- Prosimy o staranne i wyraźne zakreślenie odpowiednich ilości egzemplarzy

AMIGA VERSUS PC

czyli dyskusji ciąg dalszy

Na samym początku małe wyjaśnienie dla zwolenników i przeciwników powyższych komputerów — artykuł ten nie został napisany w celu wyraźnego wskazania, który z nich jest lepszy, lecz ma on na celu wskazanie zalet i wad powyższych produktów. Ja równie dobrze czuję się przed AMIGĄ, jak i „pecetem".

Jeśliby rozpocząć porównania od cen — to za zestaw AMIGA z monitorem czarno-białym możemy sprawić sobie co najwyżej rozbudowaną konfigurację IBM XT lub podstawową konfigurację IBM AT (jednakże z dyskiem twardym). Właściwego porównania można dokonać dopiero pomiędzy podstawową konfiguracją AMIGI z monitorem oraz PC/AT z dyskiem twardym 40 MB i kartą graficzną VGA.

Mimo, że częstotliwość zegara PC/AT jest znacznie większa (12 MHz) od Amigi (7 MHz), to przy ogólnikowym spojrzeniu na oba systemy uzyskujemy podobny zestaw możliwości. Warto też zaznaczyć na początku, że większość dobrego oprogramowania dla „pecetów" wymaga dysku twardego, co nie ma miejsca w wypadku AMIGI. Niestety sytuacja ta wygląda zupełnie odwrotnie w przypadku pamięci RAM: prawie wszystkie programy dla PC/AT można uruchomić mając standardowe 640 KB RAM, podczas gdy Amiga jest bardzo „pamięciożerna" (weźmy np. programy DTP czyli do składu tekstu).

Z drugiej strony ceny dysków twardych do AMIGI są tak wysokie, że za zwykłą dwudziestkę do AMIGI można bez większego problemu kupić setkę do PC. Otwarta architektura „pecetów" stawia je znacznie powyżej serii A500. Użytkownikom AMIGI pozostają w tym momencie albo kosztowne dodatki do wysłużonej A500, albo zamiana komputera na co najmniej Amigę 2000. Amiga 3000 jest w cenie PC/AT 486, choć (moim zdaniem) A3000 jest znacznie lepszym komputerem.

Amigi standardowo wyposażone są w system operacyjny zwany Kickstart w trzech wersjach: 1.2, 1.3, 2.0. Wersja 1.3 jest rozbudowaną odmianą systemu 1.2, natomiast 2.0 to znaczy skok jakościowy (i ilościowy — ROM zajmuje tutaj już 512 KB, zamiast 256 KB). W ROM Amigi mieści się niemal wszystko i wystarczy kilka niezbędnych plików na dysku, aby móc posługiwać się CLI (Command Line Interface — wersja DOS na AMIGĘ), lub Workbenchem (graficzny system porozumiewania się z użytkownikiem podobny do GEM na ATARI, WINDOWS na IBM, czy GEOS na C-64).

ROM PC to 8 KB — nie ma w nim prawie nic, poza generatorem znaków i procedurą inicjalizacyjną. Nie znaczy to wcale, że przez to „pecet" jest gorszy — po prostu system operacyjny jest wczytywany z dysku twardego lub dyskietki. DOS dla IBM to to samo, co CLI dla AMIGI — pod tym względem oddaję palmę pierwszeństwa „pecetom", gdyż MS-DOS 5.0 to naprawdę znakomite narzędzie do pracy.

Gorzej wygląda sprawa tzw. MS-WINDOWS — w porównaniu z amigowskim Workbenchem, czy GEMem dla ATARI ST program ten (według mnie) wygląda tak sobie, a jego komfort pracy jest co najmniej dyskusyjny. Nie przeszkodziło to jednak firmie Microsoft sprzedać prawie 6 milionów kopii tego programu...

Organizacja dysku na AMIDZE może pozostawiać wiele do życzenia zagorzałym użytkownikom IBM, ale co by było, gdyby pozbawić „peceta" dysku twardego? Gdy początkujący użytkownik AMIGI zacznie przedzierać się przez instrukcję obsługi już po kilku chwilach zrozumie co, gdzie i jak. Tak naprawdę mnogość katalogów systemowych na dysku służy tylko wprowadzeniu na nim porządku (niektóre można pominąć). W wypadku „peceta" system operacyjny może zostać umieszczony w jednym katalogu o dowolnej nazwie (z wyjątkiem dwóch plików systemowych) co niekiedy jest po prostu WYGODNE.

Odwieczny problem „zawieszania się" programów męczy rzeczywiście do tej pory użytkowników AMIGI, warto jednak w tym miejscu zauważyć, że „pecety" znajdują się już na rynku od 12 lat, a AMIGA tylko od 5. Mniej więcej tyle trzeba było czekać na porządne i guruodporne oprogramowanie użytkowe. W ciągu 12 lat PC stały się ogólnoświatowym standardem, nic więc dziwnego, że wiele programów na AMIGĘ pracuje w podobnych standardach (np. archiwizery). Istnieje także wiele emulatorów sprzętowych i programowych PC dla AMIGI, co niewątpliwie ma wpływ na popularność tego drugiego komputera jako taniego systemu zarówno do pracy jak i zabawy.

AMIGA ma także jedno doskonałe rozwiązanie techniczne, które pod tym względem stawia ją daleko wyżej niż PC — mianowicie multitasking czyli, wielozadaniowość. Pozwala ona na wykonywanie wielu programów jednocześnie (np. pisząc ten tekst na AMIDZE słuchałem jednocześnie muzyczki — nie jest to co prawda najbardziej rozsądny przykład wielozadaniowości). Nie należy tu jednak mylić DZIAŁANIA kilku programów jednocześnie z możliwością URUCHOMIENIA innego pliku z poziomu programu już wykonywanego — takie możliwości ma również pecet.

Jedną z podstawowych wad AMIGI jest to, że system operacyjny tego komputera przez jakiś czas jest odporny na wszystkie starania użytkownika, co może doprowadzić do szału (o ile masz podręcznik w bliżej niezrozumiałym języku). Pierwszy kontakt z PC jest znacznie przyjemniejszy, choć na dłuższą metę praca na obu komputerach jest kwestią przyzwyczajenia.

Dobra grafika i dźwięk to mimo wszystko ważne cechy komputera. AMIGA w podstawowej konfiguracji ma niesamowite możliwości zarówno graficzne jak i muzyczne. Pecety dość długo musiały czekać na swoją graficzną odnowę w postaci karty VGA (i SVGA), gorzej jest z dźwiękiem. Jak na razie porządne karty muzyczne do pecetów są bardzo drogie — podobnie jak specjalizowane urządzenia do Amigi (dyski twarde, przystawki graficzne itp.).

Wspaniałe możliwości graficzne i muzyczne AMIGI w pewnym sensie przyczyniły się do początkowej porażki tego komputera — powstało zbyt wiele świetnie wykonanych gier, stąd też mało kto stawiał na walory użytkowe tego komputera. Dopiero teraz można zauważyć wzrost ilości oprogramowania użytkowego porównywalnego jakościowo z „pecetami", zdecydowanie królującymi na tym polu. Powstały dobre edytory tekstu (CED, WORD PERFECT), programy DTP (PAGE STREAM), arkusze kalkulacyjne (ADVANTAGE), wiele naprawdę dobrych wersji języków programowania (ASMONE, DEVPAC, MASTERSEKA, HISPEED PASCAL 1.0, LATTICE C, AMOS). Użytkownik ma do dyspozycji, będące obecnie domeną AMIGI, programy animacyjne (IMAGINE) oraz programy muzyczne (PROTRACKER, MED, TFMX ...), które pod tym względem wyprzedzają znacznie PC.

Trzeba przyznać, że „pecety" kuszą niesamowitą liczbą programów użytkowych o najwyższej klasie — od baz danych począwszy a na programach CAD/CAE/CAM kończąc. Na polu muzyki zaś PC oferuje najbogatszą (po Atari ST) listę programów do obsługi MIDI i bije na tym polu AMIGĘ. Pod względem gier króluje AMIGA, lecz kto wie co będzie za kilka lat. Jak dotychczas, przynajmniej w zakresie gier symulacyjnych, „pecet" stoi niezaprzeczalnie wyżej. Wystarczy porównać kilka symulatorów lotu, aby przekonać się, że nawet różnica nędznych 5 MHz w pracy zegara może mieć istotne znaczenie.

Ostatnim wątkiem będzie psychologia. Nie zapominajmy, że wiele osób doradzających zakup tego czy innego komputera to ludzie, którzy po prostu mają komputer w domu. Ryzyko pomyłki jest w tym wypadku dość duże — najczęściej bowiem nasz znajomy „komputerowiec", to człowiek, który ma C-64, Amigę, ST czy peceta i o możliwościach sprzętu konkurencyjnego producenta ma blade pojęcie (oparte zwykle na ploteczkach zasłyszanych w różnych dziwnych miejscach). Zasięgając rady w ten sposób dowiesz się zatem nie tyle, który komputer jest LEPSZY, ile co ZNA Twój doradca.

Na pytanie: co lepiej kupić? — odpowiem wymijająco — AMIGĘ i peceta.

BARTŁOMIEJ DRAMCZYK

C-BASE 3.2 czyli C-64 i PROWADZENIE FIRMY

Panuje powszechne przekonanie, że do prowadzenia firmy niezbędny jest co najmniej jakiś pecet, najlepiej od 386 w górę. Co jednak zrobić w sytuacji, gdy z jakichś powodów nie chcemy kupować nowego komputera lecz raczej wolałbyś wykorzystać do tego Commodore 64?

Ustalmy od razu: nikomu nie proponuję prowadzenia księgowości, kadr i materiałówki dla dużego przedsiębiorstwa. Jeśli jednak masz tylko kilku pracowników, C-64 i stację dysków...

Do szczęścia będą Ci potrzebne dwa programy: WARSAW BASIC 3.2 i cBASE 3.2 i już masz z głowy system zarządzania bazą danych i arkusz kalkulacyjny. cBASE 3.2 (baza danych) naprawdę nadaje się do pracy. Mało tego — jest to również program, dla którego język polski jest językiem rodzimym.

cBASE 3.2 pracuje tylko i wyłącznie pod kontrolą WARSAW BASIC 3.2. Cały system zarządzania bazą danych został napisany w języku eBASE (Easy Base), który jest swoistym językiem do tworzenia tego typu programów. Składa się on z ok. 30 instrukcji. Procedury sortowania, kodowania i dekodowania bazy danych zostały napisane w kodzie maszynowym (ze względu na szybkość). cBASE sortuje 1000 pól w ciągu kilku minut! Cały program zarządzający bazę danych można wylistować. Warto zaznaczyć, że cBASE jest pierwszym systemem zarządzania bazą danych w pełni obsługującym polskie znaki. Istnieje możliwość przesyłania plików tekstowych pomiędzy cBASE, a EDYTOREM PL czyli pomiędzy bazą danych i edytorem tekstu. Innymi słowy WARSAW BASIC, cBASE i EDYTOR PL stanowią pakiet zintegrowany o dużej użyteczności (jednolity standard polskich znaków, język programowania o bardzo dużych możliwościach itp.).

cBASE 3.2 pozostawia użytkownikowi około 11 KB wolnej pamięci RAM. I tu nasuwa się pytanie: na co komu baza danych o długości 11 kB? Rzeczywiście tyle tego, co kot napłakał, lecz cBASE przeprowadza wszystkie operacje na dyskietce (z wyjątkiem sortowania).

Można założyć bazę danych o łącznej długości 600 bloków (150 kB); taka ilość bajtów wystarczy do prowadzenia małej firmy. Pozostałe 64 bloki na dysku mogą być wykorzystane jako format (deklarowany na początku) i indeks bazy danych. Taki sposób zarządzania bazą danych jest bardzo rozsądny w wypadku C-64. Każde pole w rekordzie jest bezpośrednio po wprowadzeniu zapisywane na dysk, tak więc nawet wyłączenie prądu nie straszne — wystarczy tylko wykonać operację CLOSE dla pliku bazy danych, a nie wpisywać wszystko od nowa.

Zmiana indeksu bazy danych nie powoduje ponownej reorganizacji indeksowanego pliku, ale tworzy nowy plik informujący o zmienionym indeksie dla bazy danych — umożliwia to powrót do standardowego indeksu i korzystanie z kilku indeksów naraz.

Poważną wadą cBASE jest brak możliwości sortowania danych o długości większej niż wolne 11 KB pamięci. Więc jeśli będziesz sortował dane względem pierwszego pola (np. nazwiska) to suma długości pól nie może przekroczyć 11 KB. No cóż, nie wszystko można mieć na raz, a sortowanie danych bezpośrednio na dysku to istna katorga. Drugie ograniczenie to możliwość sortowania tylko względem jednego pola numerycznego.

Jako arkusz kalkulacyjny cBASE oferuje podstawowe operacje arytmetyczne, które można wykonywać na polach numerycznych: dodawanie, odejmowanie, mnożenie i dzielenie. Wyniki zapisywane są w wybranym polu wynikowym. Nie radzę próbować na polach tekstowych tych operacji. Dodatkowym atutem cBASE jest obsługa wszystkich drukarek pracujących w oparciu o złącze równoległe (Centronics) jak i szeregowe.

cBase 3.2 ma bardzo rozbudowany mechanizm wyszukiwania informacji w bazie danych. Funkcja ta stawia cBASE niewiele poniżej poziomu profesjonalnych systemów zarządzania bazami danych.

Niestety ze względu na niewielką ilość wolnej pamięci nawet nie próbowałem stworzyć bazy danych dla mojego księgozbioru — po prostu nie byłoby jak tego posortować; nic nie stoi jednak na przeszkodzie zasiąść do języka eBASE i stworzyć własny program tego typu. Tym bardziej, że cBASE jest tylko przykładem działania języka eBASE. Jeśli już, to jest to przykład na piątkę z plusem. Istnieje także rosyjskojęzyczna wersja cBASE, nazwana rBASE (Russian Base), która operuje na cyrylicy i w tym alfabecie komunikuje się z użytkownikiem.

Bawiąc się tym programem miałem nieodparte wrażenie, że cBASE to nic innego jak mutacja bardzo popularnego w Polsce języka baz danych o nazwie dBASE. Indeksowanie, wykonywanie operacji arytmetycznych bezpośrednio na polach bazy oraz możliwość zmiany programu obsługi, to cechy programów obsługi baz danych wykraczających daleko poza możliwości C-64. Jak się jednak okazuje nie ma nic niemożliwego na świecie...

Ogólnie język eBASE to naprawdę niezły kawałek dobrej programistycznej roboty. Program ten poleciłbym użytkownikom, których nie stać na peceta czy AMIGĘ, a potrzebują dobrego, wielofunkcyjnego systemu zarządzania bazą danych. Adresuję to także do małych prywatnych firm — znany nam jest pewien warsztat mechaniczny (naprawa samochodów) obsługujący tak magazyn jak i płace właśnie za pomocą C-64 i opisywanej w tym artykule bazy cBASE V3.2.

BARTŁOMIEJ DRAMCZYK

WADY:
- mała ilość pamięci przeznaczonej na sortowanie danych;
- brak możliwości sortowania więcej niż 1 pola numerycznego;
- zbyt ogólna i chyba zbyt skąpa instrukcja obsługi;

ZALETY:
- wbudowany arkusz kalkulacyjny;
- obsługa drukarek pracujących w różnych standardach;
- wygoda obsługi;
- bezpośredni zapis danych na dyskietkę po ich wprowadzeniu

DYSTRYBUTOR:
FUNDACJA EDUKACJI TECHNOLOGICZNEJ, ul. Burdzińskiego 5, 03-480 Warszawa, tel/fax 18-01-76.

C-64 TOTAL

Niedawno udało mi się zdobyć książkę pod tytułem „C-64 TOTAL". Grube, ponad tysiącstronicowe tomisko z krzykliwą reklamą: trzy bestsellery w jednej książce. Pozycja ta jest wznowieniem trzech najlepiej sprzedających się książek wydanych przez znany zachodnioniemiecki koncern wydawniczy Markt & Technik. Stosunkowo niska cena (49 DM) niemal nakłania do kupna tej księgi (biorąc pod uwagę dosyć wysokie ceny książek w Niemczech). W Polsce udało mi się ją zdobyć za jedyne 51 tys. zł. Przejdźmy jednak do konkretów.

Pierwsza część C-64 TOTAL przeznaczona jest dla początkujących. Jest to prowadzenie za rączkę od podstaw (np. podłączenie komputera) poprzez pierwsze, krótkie programy w języku BASIC, programowanie układów SID i VIC, aż do bardziej rozbudowanych programów w tym języku. Można tu też znaleźć opis wszystkich poleceń interpretera BASIC V2.0.

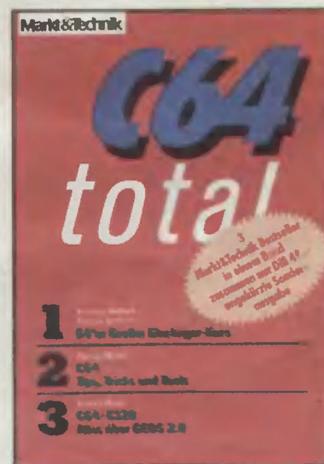
Bardziej godna uwagi jest druga część C-64 TOTAL o nazwie TIPS, TRICKS UND TOOLS. Czytelnik zostaje wprowadzony w różne tajniki C-64, począwszy od prostych zabezpieczeń programów w BASIC (np. REM SHIFT+L), poprzez opis komórek strony zerowej pamięci, proste programy w asemblerze, wykorzystywanie znanych i mniej znanych trików programowych, aż do wykorzystywania techniki okien zarówno w BASIC, jak i w asemblerze. Sądzę, że druga część C-64 TOTAL jest idealnym uzupełnieniem „DAS ANTI-CRACKER BUCH" — książki poświęconej zabezpieczaniu programów.

Ostatnia część opasłego tomiska stanowi bardzo szczegółowy opis systemu GEOS 2.0 dla C-64 i C-128. Ta część ma dosyć specjalistyczny charakter i przeznaczona jest dla ludzi korzystających z tego otoczenia w swej codziennej pracy. W połączeniu z drugim wydaniem książki MAPPING C-64 (opis dotyczący wykorzystania pamięci przez GEOS) mamy już dosłownie wszystko, co jest potrzebne do pracy z tym programem.

Sądzę, że C-64 TOTAL jest pozycją bardzo wartościową, zarówno dla początkujących, zaawansowanych, jak i użytkowników otoczenia GEOS.

BARTŁOMIEJ DRAMCZYK

C-64 TOTAL,
Markt&Technik 1991
Cena: 51 000 zł.



JĘZYKI PROGRAMOWANIA

czyli jak się dogadać z maszyną

Kolejną grupą języków programowania są języki wywodzące się od Basic Common Programming Language, czyli BCPL. Łączą one w sobie cechy języków wysokiego poziomu i asemblerów. W BCPL napisany został między innymi AmigaDOS. Niestety nic więcej nie wiem o tym języku, ponieważ nie znalazłem żadnej książki ani jego kompilatora.

C

Największą popularność z tego kręgu zdobył sobie oczywiście język C. Stworzony przez D. Ritchiego w 1972 roku został napisany z myślą o programistach, stąd też cechuje się on dużą elastycznością i zwięzłością oraz małą ilością reguł, które i tak są rzadko przestrzegane. Liczba funkcji podstawowych tego języka (w C prawie wszystko jest funkcją, łącznie z programem „głównym") jest niewielka, ale za to istnieje znaczna liczba funkcji bibliotecznych, począwszy od matematycznych, pozwalających na rysowanie wykresów, do obsługi menu czy requesterów (nawet dla IBM). Ich wykorzystanie nie nastręcza żadnych problemów, rzecz jasna tym, którzy mają odpowiednią dokumentację. Oprócz tego C ma sporą liczbę operatorów.

Kompilatory C pozwalają łączyć często wykorzystywane deklaracje i funkcje z programem w formie źródłowej za pomocą dyrektywy include, możliwe jest również ich dołączenie w postaci „połskompilowanej" do bibliotek (tylko funkcje). Biblioteki pozwalają również programiście na swobodne łączenie programów w C z asemblerem, a także swobodne (w pewnych granicach) kreowanie postaci programu wynikowego. Przejrzystość programu zależy tylko od programisty. C nie narzuca tu żadnych reguł. Dbanie o poprawność działania programu również spoczywa na programiście, bowiem C nie sprawdza, czy nie przekroczono zakresu tablicy i przy operacjach na niej i tak weźmie do obliczeń jakąś liczbę nawet, jeżeli nie będzie ona elementem tablicy. W wypadku C jest obojętne czy parametr funkcji przenosi wartość zmiennej, czy też jej adres. Takie błędy mogą się czasami skończyć komunikatem Guru.

Błędy powstałe przy wprowadzaniu programu (np. zła nazwa funkcji) często wykrywa linker, który nie może dołączyć odpowiedniej funkcji do programu. Kompilator w zasadzie sprawdza tylko, czy zgadzają się typy danych przy wywołaniu funkcji. Z tych powodów C nie bardzo nadaje się do nauki programowania od podstaw, ale jest bardzo dobrym językiem do poznania systemu operacyjnego i „wnętrzności" komputera. Zakres stosowania tego języka jest szeroki: od systemów operacyjnych (Unix, biblioteki Amigi) do obliczeń symbolicznych.

FORTH

Do tej grupy języków należy również FORTH napisany przez C.H. Moore w 1970 roku. Programy tworzone w tym języku są bardzo szybkie, i pozwalają na przeprowadzenie obliczeń w czasie rzeczywistym. Dlatego FORTH jest między innymi stosowany do pisania programów sterujących przebiegiem szybko zmieniających się procesów, ale nie tylko. Pierwszym zastosowaniem tego języka było sterowanie teleskopami astronomicznymi i przetwarzanie danych w astronomii.

Podstawową jednostką tego języka jest słowo, które odpowiada stałej, zmiennej, instrukcji czy funkcji z innych języków. Wszystkie słowa tworzą słownik na podstawie, którego tworzy się nowe słowa i tak dalej, aż powstanie cały program będący również słowem. W obliczeniach FORTH posługuje się odwrotną notacją polską i strukturą stosu. I tak, aby dodać dwie liczby np. 2+3 należy „odłożyć" liczbę 2 na stos, następnie liczbę 3, a na końcu operator dodawania. Na szczycie stosu pojawi się wynik. Z tego powodu uważa się, że język ten jest trudny do opanowania.

LISP
(LISt PROCESSOR)

Kolejna grupa języków to języki funkcyjne. Ich nazwa wynika z tego, że wywołanie funkcji jest głównym rozkazem tych języków. Pierwszym z tego rodzaju języków jest LISP. Powstał on w roku 1960, a jego twórcą jest J. Mc Carthy. Jak wynika z nazwy, LISP operuje on w oparciu o listy struktur danych (listą może być znak lub ciąg znaków). Funkcje, jakie można wykonywać na listach to przede wszystkim superpozycja, wyrażenie warunkowe i rekurencja. Pozwala to na budowę złożonych powiązań i zmieniających się w czasie relacji między listami. LISP stosuje się do pisania programów automatycznie dowodzących twierdzenia, czy wyznaczania symbolicznej postaci całek nieoznaczonych.

LOGO

Następnym językiem z tej grupy jest LOGO, napisany w Artificial Inteligence Laboratory w 1970 roku. Jest to język bardzo elastyczny. Programowanie w nim polega na budowaniu z procedur podstawowych nowych procedur. Umożliwia on również stosowanie rekurencji, czyli wywoływanie procedury przez nią samą, ale z nowymi wartościami. Ponieważ programiści wspaniale rozwiązali grafikę LOGO, znalazł on zastosowanie jako język dydaktyczny dla dzieci, do oswojenia ich z komputerem.

PROLOG

Ostatnim językiem jaki chciałbym przedstawić jest PROLOG. Należy on do nowej grupy języków logicznych. Pozwala on na postawienie zadania w postaci formuł logicznych, (a nie jak większość języków formuł matematycznych) czyli w sposób bardziej zbliżony do sposobu myślenia człowieka. Język ten jest stosowany do rozwiązywania problemów naukowych (czysto teoretycznych), budowania relacyjnych baz danych (wyszukujących i sortujących informacje według różnych kryteriów) oraz systemów ekspertowych (podejmujących decyzje na podstawie dostarczonych danych). Ten język oraz poprzednia grupa języków zalicza się do języków sztucznej inteligencji.

Oczywiście w niniejszym omówieniu nie wyczerpałem całkowicie tematu. Zaprezentowałem tylko te języki programowania, które mają jakikolwiek związek z Amigą i są ciut lepiej znane niż inne. Te które znam z własnego doświadczenia, starałem się opisać szerzej.

PAWEŁ GALAS
(TEAM ROBOCOST)

CD POD STRZECHY?

Od 1984 roku, w którym zaczęło się prawdziwe komputerowe szaleństwo w Polsce komputery bardzo spowszedniały. Początkowo nawet prasa żywo reagowała na pojawianie się coraz to nowych systemów w biurach, placówkach pocztowych czy bankach; obecnie przywykliśmy już do monitora i klawiatury, komputery wtopiły się w codzienną rzeczywistość.

Stopniowe przemiany społeczne spowodowały również wejście komputerów do domów. Nierzadko prywatni hobbyści dysponują kilkoma komputerami, zdarzają się również posiadacze sprzętu o mocy obliczeniowej wielokrotnie przekraczającej potrzeby przeciętnego śmiertelnika. Czy obecność komputera w domu jest zatem aby na pewno usprawiedliwiona?

Na to pytanie trudno dać konkretną odpowiedź. Wiadomo w każdym razie, że komputer w domu wcale niekoniecznie musi służyć wyłącznie do zabawy. Takie przekonanie panowało na początku, gdy okazało się, że osławione automaty do gier nie różnią się niczym od sprzedawanych za psie pieniądze komputerów domowych. Czy jednak każdy posiadacz 8. czy 16. bitowej maszynki musi od razu zajmować się grami?

Trzeba przyznać, że około 60% dostępnego oprogramowania dla komputerów domowych typu Commodore 64 to gry, choć nie brak w tej masie naprawdę bardzo pożytecznych i świetnie wykonanych programów narzędziowych (np. edytor tekstu Fontmaster opisywany w tym numerze "C&A"). Podobnie ale tylko podobnie wygląda sprawa z Amigą — wydaje mi się jednak, że osąd „profesjonalności" komputera na podstawie liczby wydanych dla niego gier jest kompletnie bez sensu.

Inwestowanie w komputer ma sens tylko wtedy, gdy spodziewamy się osiągnąć dzięki niemu wymierne zyski. Przykładów można podać co najmniej kilkadziesiąt — od obsługi wypożyczalni kaset video, poprzez obsługę warsztatu samochodowego do sterowania procesami technologicznymi w pewnych zakładach (Commodore 128).

Generalnie jednak przyjmuje się, że komputer nie jest wykorzystywany w domu z taką efektywnością jak pralka czy toster. W społeczeństwie amerykańskim pojawiło się swego czasu pytanie czy w takim razie kupowanie komputerów do domu jest czymkolwiek uzasadnione. Wiadomo przecież, że komputeryzowanie własnego notesu z telefonami jest po prostu bzdurą — włączenie komputera, wczytanie programu, bazy danych oraz odnalezienie zapisu zajmuje wielokrotnie więcej czasu aniżeli otwarcie kalendarzyka i dotarcie do potrzebnego numeru za pomocą skorowidza.

Ostatnio jednak malkontentom przedstawiono poważny kontrargument. Wiąże się on bezpośrednio ze stosunkowo młodą technologią produkcji dysków kompaktowych. Jak wiadomo dyski optyczne są w stanie na bardzo małej powierzchni zmieścić nieprawdopodobnie dużą ilość informacji. Jak wiadomo, jeden standardowy dysk kompaktowy jest w stanie pomieścić w granicach 600 milionów znaków. Technologia ta, znana pod nazwą CD-ROM (Compact Disk — Read Only Memory) znalazła bardzo szybko swoje zastosowanie w informatyce. Takie zagęszczenie informacji pozwala na przykład na zapis całej Wielkiej Encyklopedii Powszechnej na jednym krążku!

Możliwości te zostały już sprawdzone praktycznie i wykorzystane. Okazało się w międzyczasie, że sama encyklopedia nie zajmuje znowu tak wiele miejsca i dałoby się coś jeszcze upchnąć na takim dysku. Rozwiązanie było niezwykłe — obok encyklopedii zapisano jeszcze... utwory klasyków muzyki, reprodukcje dzieł sztuki, archiwalne zdjęcia itp. Dzięki temu człowiek szukający informacji o Beethovenie może w ramach encyklopedii posłuchać dodatkowo jego dzieł w najznakomitszym, światowym wykonaniu. Koneserzy nie ruszając się z domu mogą podziwiać obrazy Rembrandta. Nic nie stoi na przeszkodzie, aby na takim krążku zapisać trójwymiarowe wzory związków chemicznych, przedstawić graficznie przemiany zachodzące w atomach — nie ma właściwie żadnych ograniczeń.

W chwili obecnej CD-ROM stają się powoli coraz bardziej „agresywne". Jak dotychczas, główną przeszkodą był fakt, że urządzenia te pozwalały wyłącznie na ODCZYT danych. Z takiego obrotu sprawy są bardzo zadowoleni producenci oprogramowania, gdyż użytkownik będzie musiał kupić oryginalny program. Jednakże za sprawą wielu firm na rynku pojawiły się już dyski optyczne, na których można dane także zapisywać. Ma to olbrzymie znaczenie i dla producentów (tym razem osprzętu) i dla użytkowników: mając taką „dyskietkę" o pojemności powiedzmy 300 MB nie muszę kupować dysku twardego, zaś konwencjonalne stacje dysków są mi potrzebne wyłącznie do przenoszenia danych z zestawami „starszej daty". Innymi słowy mam bardzo szybki zestaw komputerowy bez dysku twardego!

Rozwój technologii poszedł w kilku kierunkach. Produkowane już są specjalizowane odtwarzacze do dysków kompaktowych, wymyślono również specjalne karty pozwalające na przyłączenie zwykłego odtwarzacza do komputera. Nie tak dawno temu oglądałem urządzenie, którego głównym przeznaczeniem była współpraca z komputerem; po wyłączeniu komputera można było używać odtwarzacza jako zwykłego „kompaktu". Nieco droższym rozwiązaniem jest choćby CDTV firmy Commodore w którym, oprócz samego CD, znajduje się również komputer.

Do niedawna „wyłączność" na korzystanie z dobrodziejstw CD mieli wyłącznie posiadacze peceтów i innych „profesjonalnych" maszyn. Niedawno jednak monopol ten został złamany przez firmę Commodore, która nie tylko wyprodukowała wspominane już CDTV, lecz sprzedaje też nowe CD dla Amigi — A570 (patrz strona 9). Czekamy zatem na odpowiednie urządzenie lub interfejs dla „małego" C-64...

KLAUDIUSZ DYBOWSKI

Nowa generacja pamięci masowych — CD-ROM A570 dla Amigi. Szczegóły na stronie 9.